VERTEX SPACE ANALYSIS FOR MODEL-BASED TARGET RECOGNITION

Final Report

Contract #N00014-95-1-0859

August 1996

Submitted to:
Office of Naval Research - 1264
800 North Quincy Street
Arlington, VA 22217-5000
Attention: Dr. William Miceli

Submitted by:
Center for Automation Research
University of Maryland
College Park, MD 20742-3275

Principal Investigators:

Azriel Rosenfeld
Email: ar@cfar.umd.edu
Phone: (301) 405-4526
Fax: (301) 314-9115

Gary Whitten
Email: whitteng@cfar.umd.edu
Phone: (301) 405-1736
Fax: (301) 314-9115

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
August 1996

3. REPORT TYPE AND DATES COVERED
Final Technical Report

4. TITLE AND SUBTITLE
Vertex space analysis for model-based target recognition

5. FUNDING NUMBERS
N00014-95-1-0859

6. AUTHOR(S)
Gary Whitten and Azriel Rosenfeld

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3275

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research—1264
800 North Quincy Street
Arlington, VA 22217-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Navy position, policy, or decision, unless so designated by other documentation.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release.
Distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

We have developed powerful ATR technology that effectively addresses the critical issues of battlefield operation. Inherent in our ATR approach is a treatment of partially occluded, degraded target signatures, clutter and sensitivity to distinguish similar targets. We achieve this capability through judicious choice of features and feature attributes, local feature extraction techniques, and an efficient search strategy that exploits invariant feature attributes using our Vertex Space representation.
We use low-level orientation processing combined with consistency reasoning to extract stable features and properties, used as feature attributes. Robust recognition is accomplished by matching sensor features with like model features through the rich hypothesis prediction and verification paradigm of Model-Based ATR. We achieve tolerance to clutter by using an efficient search strategy performed in our unique invariant representation, Vertex Space, that reduces both the dimensionality and size of the required search space. Vertex Space mapping results in a reduced representation that serves as a characteristic target signature which is invariant to four of the six viewing geometry degrees of freedom. By conducting the initial search in Vertex Space, the search process is significantly simplified. We have demonstrated model-to-image feature matching using real IR imagery and models of actual military targets.

14. SUBJECT TERMS
ATR, feature matching, infrared imagery, model-based target recognition, vertex space

15. NUMBER OF PAGES
51

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements.***

Block 1. Agency Use Only *(Leave blank).*

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	PR	- Project
G	- Grant	TA	- Task
PE	- Program Element	WU	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- DOD - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- DOE - See authorities.
- NASA - See Handbook NHB 2200.2.
- NTIS - Leave blank.

Block 12b. Distribution Code.

- DOD - Leave blank.
- DOE - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- NASA - Leave blank.
- NTIS - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only).*

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

INTRODUCTION

A major goal of the Office of Naval Research (ONR) is to extend the performance of current Automatic Target Recognition (ATR) technology to enable the development of systems that can be useful in real world battlefield conditions. To be of practical use in battlefield conditions, an ATR system must be both reliable and accurate to achieve lethality while reducing risk of collateral damage. In particular, there is a need for high level discrimination so that similar targets can be distinguished to avoid fratricide. Further, an effective ATR system must be operational over a wide range of engagement approaches, weather, thermal, lighting, and terrain conditions and must be capable of performing well even when target sensor data is significantly degraded due to poor visibility, or occlusion, and should be resistant to highly cluttered environments. The required ATR system performance must be achieved within the power, speed, size and cost constraints dictated by real world battlefield conditions.

While battlefield conditions often conspire to reduce the fidelity of target signatures (e.g. reduced contrast, occlusion, broken contours), high clutter environments can greatly burden ATR processing, making it very difficult to extract degraded signatures without becoming overwhelmed by data. Highly textured scenes and urban environments can both induce high clutter conditions into an ATR scenario. Therefore, successful ATR systems must be inherently efficient to operate effectively in all required environments.

We are sensitive to these practical ATR system requirements and the associated technical issues and have been working with the Office of Naval Research (ONR) on recent programs, #N00014-92-C-0087 and #N00014-95-1-0859, to develop ATR technology appropriate for Visible and Infra-Red (IR) systems that effectively addresses these issues [Whi95].

Our ATR approach achieves tolerance to degraded and/or incomplete target signatures by using local features that are robustly extracted without need for finding intermediate high quality continuous edges or regions. Instead, our approach uses low level orientation processing combined with consistency reasoning to extract stable features and properties, used as feature attributes. Low level processing alone, such as high frequency filtering, tends to extract too many false features because of numerical instability in the vicinity of high frequency image information, which is often where features (e.g. corners) of interest occur.

Robust recognition is accomplished by matching sensor features with like model features through the rich hypothesis prediction and verification paradigm of Model-Based ATR (MBATR). We achieve tolerance to clutter by using an efficient search strategy that is directed by feature attributes and performed in our unique invariant representation, Vertex Space, that reduces both the dimensionality and size of the required search space [Whi88]. Vertex features (curvature maxima) are extracted from images and mapped to points in Vertex Space based on their local, invariant properties. The Vertex Space mapping results in a reduced representation that serves as a characteristic target signature, which is invariant to four of the six degrees of freedom associated with three dimensional (3D) viewing geometry. By conducting the model to image search, initially, in Vertex Space, the search process is significantly simplified.

The extracted features and their attributes are used to index into a model database. To generate the model database, 3D geometric and sensor properties of the model are automatically and systematically encoded into a set of compact 2D representations, where each representation is derived from a single synthetic view of the object by mapping it to Vertex Space. Our model database efficiently captures essential, invariant structural target information of complex target models, achieving a principal, and elusive goal of Model-Based ATR.

Features from sensor data are analyzed in a similar way and their invariant properties, as represented in Vertex Space, index into the model database and direct the search process, yielding hypotheses consisting of candidate representations, their associated two dimensional (2D) views and potential correspondences of model features with sensor image features. The generated hypotheses are quickly verified or rejected by projecting model features into the image using the candidate viewing geometry, and determining the level of correspondence.

In our previous ONR programs, a high level search strategy exploiting Vertex Space was developed and its computational complexity was analyzed theoretically and with automated clutter experiments. Indexing (rapid preliminary recognition) results were demonstrated for simple and complex real targets using target models, synthetic imagery and randomly generated image clutter. In addition, techniques were developed, based on implicit contours derived from orientation consistency, to extract stable features from real FLIR imagery for Vertex Space analysis, (see "Automated Missile Aim Point Selection Technology Final Report" for ONR contract #N00014-92-C-0087).

In our recent ONR program, #N00014-95-1-0859, we evaluated our system performance using additional IR imagery (from WPAFB, NVEOL and Eglin Air Force Base) and consistent models and found that the sensitivity to small scale image details was limited by the nature of extracted curvature regions. Increased sensitivity to accommodate low resolution imagery typically results in interpreting much of the image clutter as valid features. Further, curvature regions associated with small target features often interact, resulting in regions merging into amorphous blobs. Since system performance is limited by the quality of extracted features, improved feature extraction, specifiically curvature region processing, was identified as a critical need.

We developed and implemented effective curvature processing techniques, including non-linear filtering and region splitting based on curvature strength and feature orientation, which greatly improved our system's feature extraction performance. Our system is now capable of reliably resolving densely packed, interacting, small scale features. As a result, we were able to demonstrate feature matching between real IR target imagery and the corresponding model, using only target geometry information. We have not yet explicitly implemented synthetic IR signature generation.

Our robust local feature extraction approach together with our efficient search strategy provides the foundation for a powerful ATR system that can deliver the demanding level of performance required by real battlefield conditions. Our ATR system currently consists of feature extraction, indexing and intermediate feature matching, and model database generation modules. These modules can be used independently or together to form the core of a complete end-to-end ATR system with the promise of extending the operational range of ATR with respect to reliability, sensitivity, signal to clutter, occlusion, obscuration and adverse viewing conditions. Our programs with ONR have succeeded in evolving our ATR technology towards the realization of an ATR system that is capable of practical performance in the demanding battlefield environment.

TECHNICAL APPROACH

To accommodate incomplete and/or degraded target signatures (due to occlusion, poor visibility and other adverse conditions), a necessary capability for achieving practical ATR system performance, and to avoid the difficulties of region segmentation, we use local features, specifically, vertices which, conceptually, are instances of intersecting straight lines. The vertex features are incorporated into a model-based ATR framework that is appropriately sensitive to the information contained in these features. The model-based approach to ATR has shown much promise in achieving high levels of target discrimination of reduced target signatures in highly cluttered environments (as required for target identification sufficient for distinguishing between two similar targets in the same class, e.g., friendly tank from enemy tank in battlefield conditions). The model-based approach derives target information from target and sensor models, so the target information is not limited by an unwieldy training set of images and the tedious and unreliable training task is not required. The model-based approach achieves robustness of recognition by finding an optimum match between target models and image data through a powerful hypothesis prediction and verification scheme (see Fig. 1).

However, the powerful MBATR techniques come at the expense of potentially high computational complexity due to the required search through the space of model instances for the best match with observed data. An undirected and unconstrained search performed directly in the image domain cannot be implemented efficiently enough to be

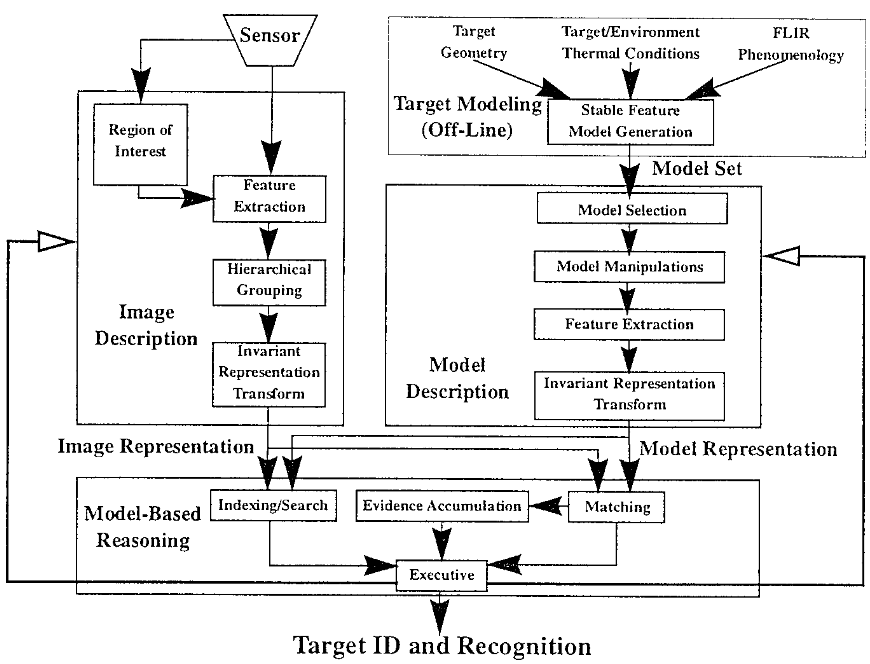


Fig. 1) MODEL-BASED ATR SYSTEM

of practical use in real smart weapons systems. To perform direct matching using feature locations alone is a problem of $O((N_i*N_m)^3)$ complexity, where N_i is the number of observed image features and N_m is the number of model features. Typically, N_i could be 1000 and N_m could be 100, resulting in a problem of immense computational complexity. However, we have developed a highly efficient directed search strategy using an invariant representation (a representation that is stable over some useful range of viewing conditions, such as geometry and thermal properties, in the case of IR imagery) that reduces and constrains the required search space such that the full power of MBATR can be realized in a practical implementation.

The first stage of our search uses local, invariant properties of extracted vertex features, such as angular size and relative orientations, to index into the model database. The 3D geometric properties of the model are automatically and systematically encoded into a set of compact 2D representations, where each representation is derived from a single synthetic view of the object (using appropriate sensor models), which solves many of the problems commonly associated with the efficient creation and manipulation of model databases for complex targets. Features from sensor data are analyzed in a similar way and their invariant properties index into the model database and direct the search process, yielding hypotheses consisting of candidate representations, their associated 2D views and potential correspondences of model features with sensor image features. The generated hypotheses are quickly verified or rejected by performing efficient matching between candidate sensor and model features, which provides a powerful prediction, verification mechanism that is highly effective for avoiding false target declarations.

Geometric Reasoning and Model Manipulation

To avoid the unsolved problem of extracting geometric features directly from the 3D model, which, if done rigorously, involves exploring all 3D surface regions in all directions, we generate a database of representations derived from the systematic generation of synthetic 2D images using the appropriate sensor model. Each image is associated with a different 3D view of the target object. The different views that contribute to the model database can be associated with points uniformly sampled from a sphere surrounding the model object where each point defines a location from which to view the object. We have developed C software modules, using X Windows and Silicon Graphics OGL (Open Graphics Library) graphics calls (all widely used standards), to perform the necessary 3D model manipulation and rendering. Our system automatically generates the required synthetic views (as determined by user specified parameters that control geometry resolution), extracts features and creates the model data base for a given target.

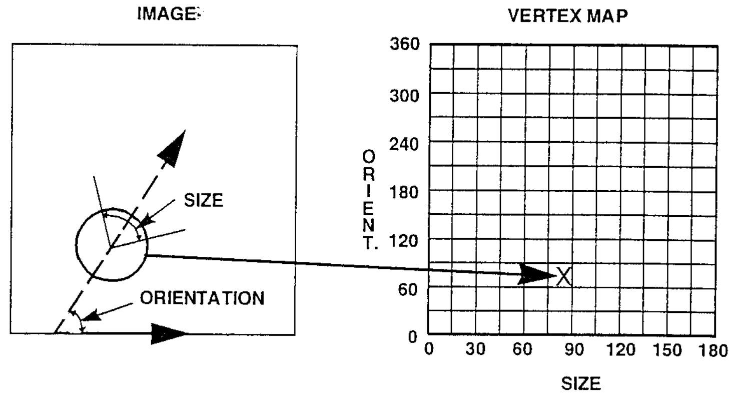


Fig. 2) MAPPING TO VERTEX SPACE

Indexing

Indexing generates the hypotheses to be investigated. Indexing is the initial prediction of target, viewpoint, and correspondence between model and sensor features based on the most significant features extracted from the sensor image data. In general, a viewpoint hypothesis is generated based on an assumed specific correspondence between image and target features, which defines the viewpoint.

To index into the view database, we use Vertex Space, our unique representation, that decreases the size and dimensionality of the search space required for model to data matching by exploiting the invariant properties of vertices extracted from images [Whi88].

An extracted vertex (intersecting straight lines or contour segments of maximum

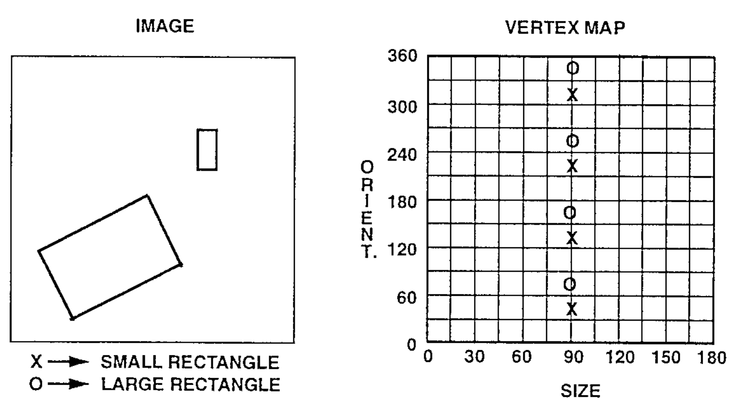
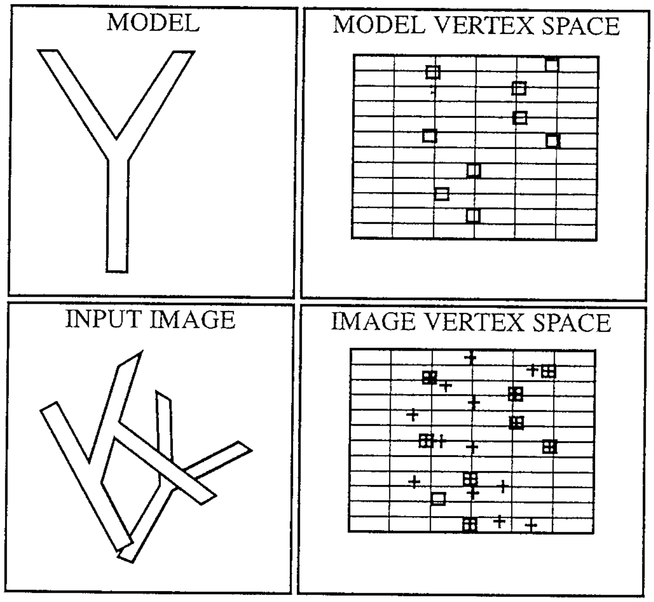


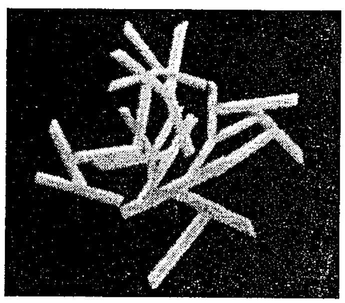
Fig. 3) INSENSITIVITY OF VERTEX SPACE TO TRANSLATION, SCALE AND 2D ROTATION



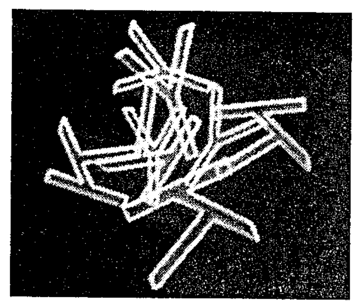
1 TEMPLATE
for
VERTEX SPACE MATCHING
vs.
250 TEMPLATES
for
IMAGE DOMAIN MATCHING

(based on 200% scale variation
1 template per
10% scale variation
1 template per
10 degrees rotation)

Vertex space object recognition process. For 2D objects, recognition is reduced to simple template matching. (Note the occlusion of the center "Y" vertex.)



Input image consisting of multiple, occluding 2-D objects ("Y" and "K") in various sizes and orientations.

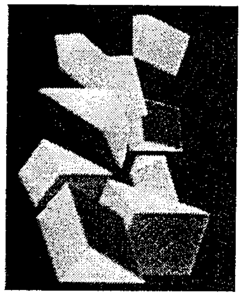


Recognized objects are highlighted by superposing projections of the model objects.

Fig. 4) 2D OBJECT RECOGNITION USING VERTEX SPACE

curvature) can be essentially described by its location and orientation in the image, as well as the size of the angle subtended by its intersecting tangent lines. Although the spatial location of a vertex is highly sensitive to image formation details, its orientation and angle size are invariant to many of these details but can still characterize essential object structure. Therefore, our vertex representation emphasizes the two invariant properties, angular orientation and size. A vertex in image space maps to a point in Vertex Space as determined by its angular size and orientation, independent of its spatial location in the image (see Fig. 2).

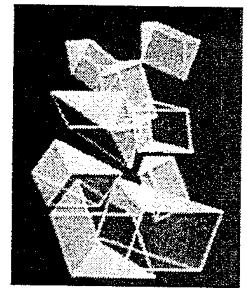
A Vertex Space mapping is insensitive to changes of scale, translation, and rotation



Multiple occluding 3D solid objects in various orientations and sizes.

GEOMETRIC DEGREES of FREEDOM:

VERTEX SPACE - 2 vs. IMAGE DOMAIN - 6



Recognized objects are highlighted by superposing projections of the model objects.

Fig. 5) 3D VERTEX SPACE OBJECT RECOGNITION

within the image plane so it serves as a viewpoint insensitive characteristic signature for the object. Since Vertex Space is affected only by changes in vertex size or orientation, Vertex Space is clearly invariant to image translations and changes in scale. Image plane rotation results only in a constant shift in the orientation dimension of Vertex Space. But, since orientation is defined with respect to an arbitrary axis, only relative orientation is meaningful (see Fig. 3).The insensitivity of Vertex Space to many of the details of two dimensional imaging geometry (a useful approximation for high altitude imaging and objects confined to a plane) results in enormous recognition simplification. For two dimensional recognition, Vertex Space provides an invariant signature for targets and the matching process reduces to simple template matching in Vertex Space with no need for hypothesis generation, since all Vertex Space representations are equivalent (see Fig. 4). In Vertex Space, a single template is sufficient, while If performed using correlation in the image domain, hundreds of templates would be necessary to cover the required range of scale and orientation. In addition, the approach is robust with respect to corrupted or occluded information since each vertex contributes independently to matching confidence, unlike techniques that depend on reliable segmentation of signature silhouettes. The usefulness of Vertex Space invariance extends to the more general three dimensional recognition problem (see Fig. 5).

Three dimensional imaging geometry is characterized by six degrees of freedom, three associated with translation and three associated with orientation. Without loss of generality, we can define a composite rotation about three orthogonal axes, such that the last rotation is about the optical axis. The advantage of such a coordinate system is the optical axis rotation is equivalent to rotation within the image plane and produces only a simple constant offset in Vertex Space, which does not affect target signatures. Of the six spatial degrees of freedom, only the two associated with rotations about axes orthogonal to the optical axis have a meaningful affect on a Vertex Space representation of an object.

The corresponding reduction of the search space greatly simplifies the search process for three-dimensional model to data matching. Matching in Vertex Space provides a complete specification of viewing geometry (by determining correspondence among three or more model and image points) so that final matching can be performed by mapping the entire model object into the image. Vertex Space is used for highly efficient preliminary matching, or prescreening, and only when a match in Vertex Space is found is the more costly image space matching performed. Like the two dimensional recognition approach, described above, three dimensional model-based recognition using Vertex Space greatly simplifies the initial matching process and provides robustness with respect to occluded or corrupted data.

In the application of Vertex Space to 3D ATR, the 3D geometric and sensor properties of a target model are automatically and systematically encoded into a set of compact 2D Vertex Space representations, where each representation is derived from a single synthetic view of the object (using appropriate sensor models), which solves many of the problems commonly associated with the efficient creation and manipulation of model databases for complex targets. Features from sensor data are analyzed in a similar way and their invariant properties index into the model database and direct the search process, yielding hypotheses consisting of candidate representations, their associated 2D views and potential correspondences of model features with sensor image features. The generated hypotheses are quickly verified or rejected by performing efficient matching between candidate sensor and model features, which provides a powerful prediction, verification mechanism that is highly effective for avoiding false target declarations.

SEARCH STRATEGY

Our emphasis in the development of techniques to perform model to image matching is efficiency with respect to clutter and robustness with respect to missing or corrupted image data. An important issue, closely related to computational efficiency, is the matching mechanism. Matching 2D image data directly to 3D models is a difficult problem since it requires detecting all potential 3D features that could project to a 2D feature, which, if done rigorously, requires examining regions, in all possible directions, about every surface point. Even if these points could be effectively found directly in the 3D domain, many more features than necessary would have to be considered simultaneously, since only a small subset is typically visible from any single view (e.g., all edges of rectangles that approximate a cylinder must be considered as possible matches with a straight line in an image, even though, typically, only two will ever be visible at the same time). In contrast, it is straightforward to match 2D critical points extracted from images directly with 2D projections of the model. For this reason, we generate a database of views of a given 3D target model and perform matching with the set of views rather than the model itself.

To avoid the complexity of exhaustive image to model feature matching, we use feature attributes and the invariance of Vertex Space to direct a multi-stage search process. The search strategy is hierarchical where viewing geometry and feature grouping is constrained incrementally so that each search level performs a small subset of the whole search, greatly reducing the combinatoric complexity, and subsequently reducing the data that must be considered at the next stage (notice the data flow structure in Fig. 13).

The first stage of the search exploits the invariant properties of Vertex Space to efficiently index into a model database of views using a hash table indexing approach which is an excellent match with Vertex Space since hashing techniques require a binary array representation, that Vertex Space naturally provides. The result of Vertex Space indexing is candidate model views and image to model feature correspondence. The hypotheses generated by indexing are then checked for orientational consistency, which determines an offset angle of image rotation. Surviving candidate features are then evaluated to determine positional consistency, which is the first stage of the search process that uses any feature location data. Positional consistency is based on predicted orientations of lines defined by feature pairs, so scale information is not needed. The surviving features are finally checked for scale consistency. The result is a correspondence between image features and model features as well as a full approximate description of the viewing geometry.

The model to image feature correspondence can then be used to refine the viewing geometry which, in turn, can be used to map the complete target model into the acquired image for final correpondence determination. The final stage involving analysis of the complete target model is costly, but is only performed when there is high confidence that a target is present as a result of the efficient feature matching search. The constraints enforced by the feature matching process ensure that false alarms will be unlikely. It can be readily seen that, while our approach is efficient, each step of the search process is directed by the results of previous steps, it can also accommodate missing features since the search process is valid for any subset of features. We describe the details of our search process in the next section.

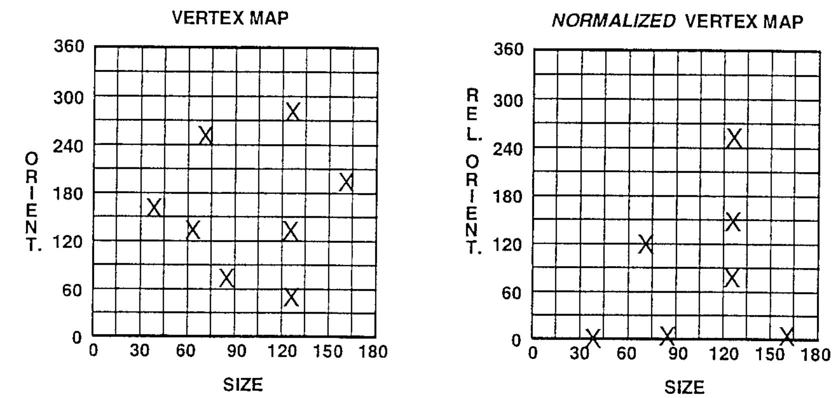
Search Approach: Curvature Directed Search Using Vertex Space

Our multistage search process begins with indexing. Indexing is the initial prediction of target, viewpoint, and correspondences between model and sensor features based on the most significant features extracted from the sensor image data. Indexing generates hypotheses to be investigated. Specifically, viewpoint hypotheses are generated based on candidate correspondences between image and target features. In addition to viewpoint, other conditions affecting target signatures are considered, e.g., thermal properties via the appropriate sensor models.

The off-line preparation of an indexing database requires extensive manipulation and

Fig. 6) NORMALIZED VERTEX SPACE

record ***RELATIVE*** pairwise difference orientations of vertices in same size bin instead of absolute orientations



analysis of the target model. However, real target models can be highly complex making manipulation difficult. To avoid the difficulties resulting from 3D target model feature analysis, we produce a compact database of Vertex Space representations derived from the systematic generation of synthetic views of the target model, using the appropriate sensor model. Each 2D image generated is associated with a different 3D view of the target object. The different views that contribute to the model database can be associated with points uniformly sampled at some specified resolution from a sphere surrounding the model object where each point defines a location from which to view the object.

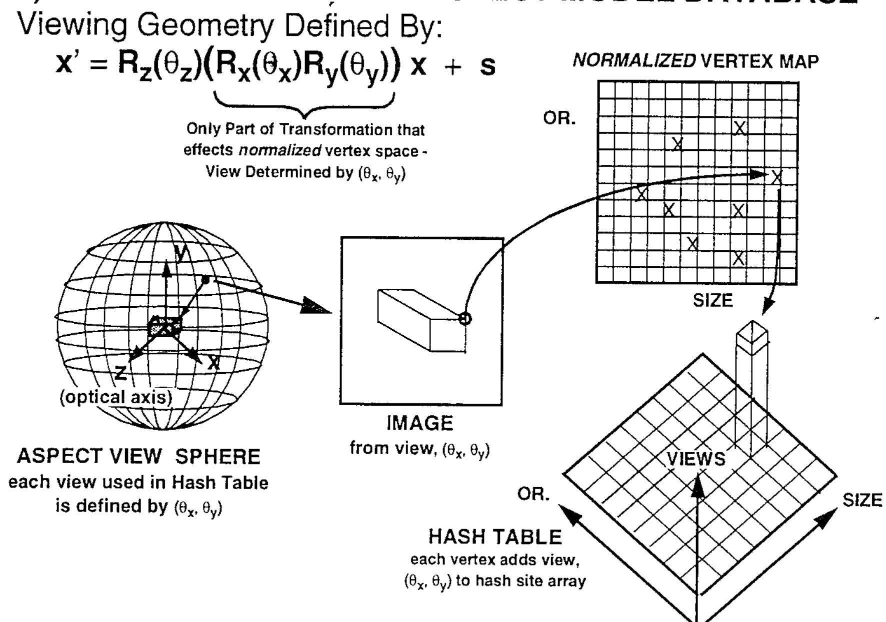
If not for the invariance of Vertex Space, we would also need to consider points of varying distance along rays centered on the object (spheres of different radius) and, at each point, different rotations about the optical axis. However, the translation (including scale) and rotational invariance of Vertex Space requires that only a single view from each point on the tessellated sphere be generated.

Analytically, the viewing geometry transformation that maps a point, whose location is determined by the 3D position vector, **x**, to the transformed point, **x**', is described by the 3D transformation

$$\mathbf{x}' = \mathbf{R}_z(\theta_z)(\mathbf{R}_x(\theta_x)\mathbf{R}_y(\theta_y))\,\mathbf{x} + \mathbf{s}$$

where $\mathbf{R}_k(\theta_k)$ represents a rotation of θ_k about axis **k**, and **s** is a translation vector. Vertex Space is invariant with respect to **s** and will only be shifted by a constant amount, θ_z, along the orientation axis. To achieve complete invariance with respect to $\mathbf{R}_z(\theta_z)$ (rotation within the image plane), we normalize Vertex Space by mapping all pairs of vertices within the same size bin to Normalized Vertex Space bins based on the average size of the vertex pair and a *relative* orientation, found by taking the difference between the two vertex

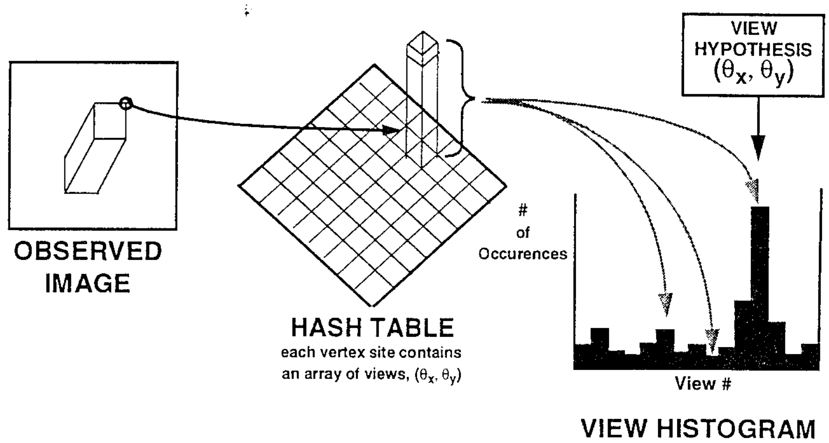
Fig. 7) BUILDING HASH TABLE ASPECT MODEL DATABASE



orientations in the pair (see Fig. 6). Normalized Vertex Space removes the dependence on orientation offset by considering only orientation differences. Forming vertex pairs results in an increase in the number of features (N -> N * (N-1), where N is the number of vertices in a size bin) that must be analyzed, but relative orientation information can be used to realize much more distinctive Vertex Space signatures than size information alone, resulting in more robust indexing, which increases the efficiency of the subsequent search stages. We form pairs based on vertices with similar size attributes since many targets of interest have symmetric features. When symmetric features undergo viewing transformations, they tend to maintain size similarities. Forming pairs from all possible feature pairs would result in too many features. In dense data domains (highly complex targets, high resolution imagery or high clutter) it may be more appropriate to use standard (unnormalized) Vertex Space and perform indexing with explicit quantized orientation offsets on single vertices to avoid forming too many vertex pairs. The details of determining which approach to use, single vertices and absolute orientation in standard Vertex Space or vertex pairs and relative orientation in Normalized Vertex Space, will be developed in the complexity analysis below.

We proceed by synthetically generating each of the views defined by the tessellated sphere. We map the resulting synthetic image to *normalized* Vertex Space and, for each vertex, we enter the current view index number into a hash table at the corresponding normalized Vertex Space site so that the hash table accumulates all the view indices that are associated with each observed vertex in Vertex Space (see Fig. 7). As mentioned earlier, by using an aspect hashing graph to represent geometric object information, we avoid one of the more difficult problems associated with model manipulation, which is the determination, from the model, of visible features. Rather, we generate all required views

Fig. 8) 2 STAGE INDEXING - First Stage:
Using **HASH TABLE** to find View Paramaters, (θ_x, θ_y)



explicitly, off-line, so the visibility of edges is determined via the synthetic image generated for each view used to build the hash table. We have automated the hash table generation process so the hash tables can be generated for any object that has a model in our system and both view angle and hash table angle resolutions can be specified. We perform hash table generation, model manipulation, synthetic image generation, algorithm design, implementation and evaluation on our MB-ATR system testbed which uses the C, Unix, X Windows and Silicon Graphics GL graphics standards.

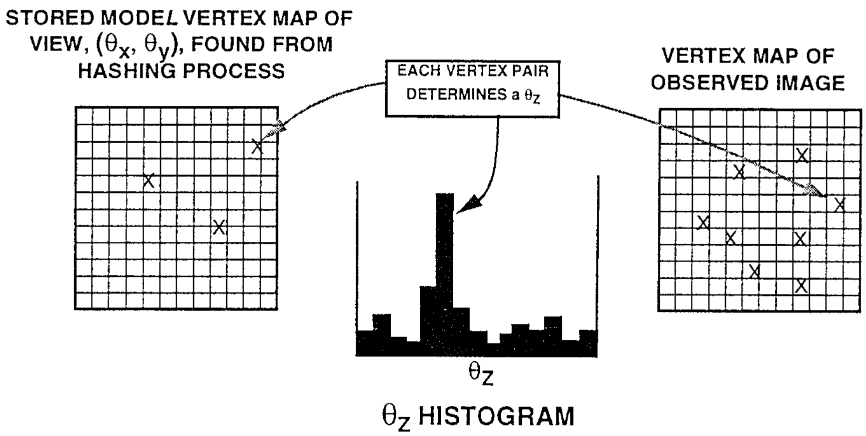
We employ a two-stage indexing process using the hash table to determine viewpoint and model to image vertex correspondence. In the first stage the θ_y, θ_x parameters are found by mapping the observed image to normalized Vertex Space. For each vertex mapped, the corresponding hash table site is accessed and the set of views stored at the site are distributed into a viewpoint histogram. The resulting peaks indicate candidate views (see Fig. 8).

The θ_y, θ_x view parameters found from the first stage of the indexing process are used to identify a candidate model Vertex Space map, which is compared to the image Vertex Space map. Sets of potentially corresponding model and image vertices define an index into θ_z values, the final parameter required to completely specify object orientation. Specifically, we recover θ_z and the model to image vertex correspondence from the second stage of indexing. The hash table indexing provides a (θ_y, θ_x) hypothesis and associated model Vertex Space representation which is compared to the observed image Vertex Space representation, which is *not* normalized (normalization is only required for the first stage of indexing). A correspondence between a model and image Vertex Space entry defines an offset angle, θ_z. The resulting value is accumulated in a θ_z histogram and

Fig. 9) TWO-STAGE INDEXING - Second Stage:

Find θ_Z (Orientation Offset Angle)

- process determines model to observed vertex correspondence
- vertex correpondence used to find all viewing geometry



any peaks indicate candidate θ_z values (see Fig. 9). Once a model Vertex Space map is identified, additional constraints can be derived from positional properties of the object view, which can verify or reject the working (θ_y, θ_x, θ_z) hypothesis and further constrain the viewing geometry and model to data vertex correspondence, as described below.

Our indexing scheme requires mapping continuous data into discrete grids so it is subject to quantization error, where vertex entries straddle bin boundaries, which can seriously degrade performance. The quantization error is aggravated by smaller bin sizes, which introduce more bin boundaries, thereby increasing the likelihood that vertex entries will fall near bin boundaries. To solve this problem, we distribute hash table entries about the single bin determined by the quantized values. A table entry is placed in the discrete bin it normally falls into, and is also placed in its (three) nearest neighbors bins with weights determined by the distance from the desired continuous value of the table entry to the center of each bin (see Fig. 10). This ensures that angular values that straddle bin borders still register, even if they should cross over the bin boundary, which allows for increasing the bin resolution arbitrarily without concern for adverse quantization effects. This approach has effectively removed all problems associated with discrete quantization error.

The result of the indexing process described above is a candidate (θ_y, θ_x) view, and θ_z optical axis rotation and candidate correspondence between model and image features, which typically identifies a small subset of all image features. The (θ_y, θ_x, θ_z) hypothesis is equivalent to defining a properly rotated (in the image plane) 2D view of the model object. It is still necessary to determine the translation and scale transformation (a

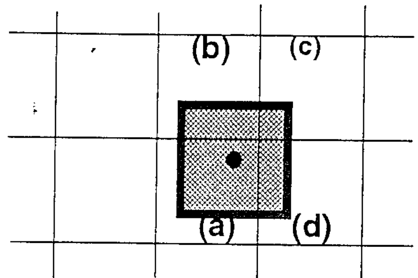


Fig. 10) CONTINUOUS HASH TABLE ENTRIES

An area the size of a hash bin (indicated by cross-hatching) is centered around the continuous hash table value (indicated by the dark circle). A value is stored in each hash bin equal to the size of the overlap area. In this case, bin a would receive the highest entry value, followed, repectively, by b, d and c.

subset of the general affine 2D transformation) required to align the hypothesized view with the observed data. Determination of the required 2D transformation requires matching an image pair of features with a corresponding pair of model features (assuming each feature has a well defined point location). Verification of the hypothesized 2D transformation requires at least one further model to image feature pair correspondence. So the geometric matching problem, after the indexing stage, reduces to a pairwise search.

We employ simple geometric constraints to determine and verify the 2D transformation necessary (along with $(\theta_y, \theta_x, \theta_z)$) for fully defining the viewing geometry.

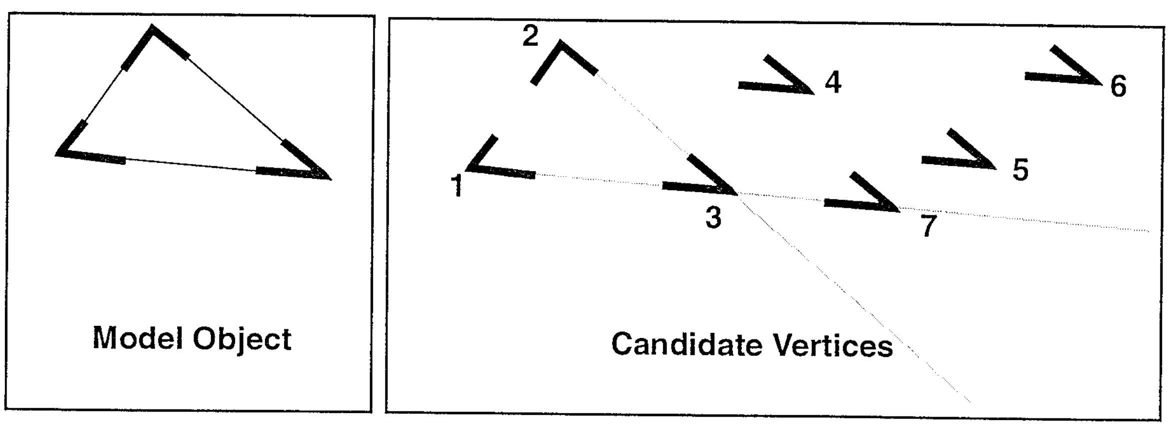
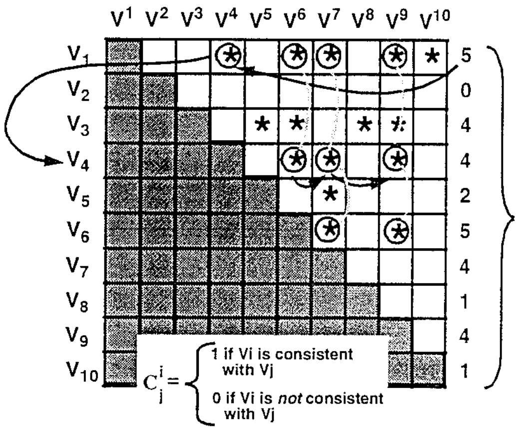


Fig. 11) POSITIONAL CONSISTENCY IN θ_Z INDEXING

Based on the Vertex Map derived from (θ_y,θ_x) indexing alone, image vertices 1 and 2 and any of 3, 4, 5, 6, or 7 could be potential matches with the model object. However, if we incorporate simple positional information that could be stored with the vertex data, the ambiguities could be readily resolved. Specifically, the angle $\overline{13}$ makes with $\overline{23}$ would reduce the possibilities for the third vertex to either 3 or 7. When the additional constraint imposed by $\overline{32}$ with $\overline{12}$ is considered, only one choice (#3) remains. No scale information is required.

Fig. 12) CONSISTENCY MATRIX, C - Drives Pairwise Search:

- **C** stores binary results of all pairwise consistency checks
- look for vertices that are consistent with a common vertex (look for matrix rows with common elements, candidate rows determined by elements in prior rows)
- choose reference index based on most consistent pairings, T_m



V4, V6, V7, V9 are mutually consistent

★ pairwise consistent

⊛ verified pairwise consistency, (consistent with 2 or more vertices)

T_m, # of Consistent Pairings for vertex m

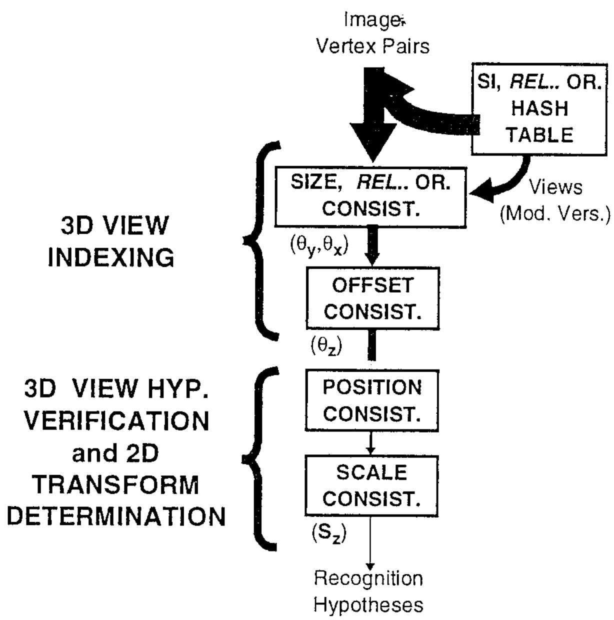
$$T_m = \sum_{k=1}^{m-1} C_k^m + \sum_{k=m+1}^{N} C_m^k$$

(for N vertices)

While the indexing process deals exclusively with local properties of individual vertices (e.g., angle size and orientation), the remaining search uses vertex positions. Without knowing scale, we can infer, from the model, the orientation of a line connecting any two corresponding image vertices. Therefore, one candidate image vertex constrains each other candidate image vertex to a specific line. Once an additional candidate vertex is found to lie on the line specified by the first, the two constitute a consistent candidate vertex pair that completely constrain the locations of all other candidate vertices and define the viewing geometry (by determining the previously unknown 2D translation and scale transformations - see Fig. 11).The viewing geometry specified by such a base pair can then be verified by the remaining candidate image vertices.

To perform this pairwise search and verification process efficiently, we do a breadth first determination of all possible pair consistencies and store the binary ("1" for consistent, "0" for not consistent) results in a consistency matrix, **C**. The remainder of the search can then be highly directed by geometric consistency constraints. The total number of potential mutually consistent candidate vertices, T_m, associated with a single reference vertex is the sum of all "1" entries in all matrix elements associated with the reference vertex. The search starts with the single reference vertex having the highest number of potential matches, as indicated by T_m. All "1" entries in the base reference row are consistent with the reference vertex. Each of these entries can be used to reference another row. All entries common to these two rows are mutually consistent and satisfy all

Fig. 13) HIERARCHICAL, DIRECTED SEARCH STRATEGY



- **Search strategy decomposes search into simpler search components**
- **Each component implements simpler search and further reduces data to be analyzed by next stage**

geometric constraints. Therefore, we select a first reference row based on the T_m values. For each entry in this row, the corresponding row is checked for common entries, which must be mutually consistent. This process is repeated, for descending T_m values until the minimum required number of verified image vertices are found or until there are no more rows with T_m values equal to or greater than the minimum required match number (see Fig. 12).

Once the image set of vertices that matches with the model object has been identified and the viewpoint determined, the model object can be mapped into the image for a final image domain correspondence determination, fully verifying or rejecting the Vertex Space hypothesis.The matching vertices that contributed to the view determination can be used to find a least squares solution that will precisely refine the viewing geometry estimate found in the matching process described above.

Our overall search strategy decomposes the computationally intensive end-to-end model to image matching search into smaller search components. Each of these is far simpler and computationaly more tractable than the global search as well as further constraining and determining the imaging geometry and establishing candidate image to model vertex correspondences. These search components are hierarchicaly related such that each level further reduces the amount of image data that must be dealt with by the next level (as represented by the line widths in Fig. 13) in a "divide and conquer" strategy.

Feature Based Matching Complexity Analysis

To analyze the complexity of our model to image feature matching search strategy, we make the distinction between the first module of the search, indexing into the 3D view and the second module, which verifies (or rejects) the 3D view hypothesis and determines the appropriate 2D transformation which includes both scale and translation information (see Fig. 13). In the following analysis, VS is Vertex Space, and *we make the assumption that vertices tend to be uniformly distributed over Vertex Space*, which of course is not strictly true, but, from subjective observation of clutter experiments, it appears to be a reasonable approximation.

$\mathbf{N_i}$ is the number of Image Vertices, $\mathbf{M_i}$ is the average Image VS bin Population

$\mathbf{N_m}$ is the number of Model Vertices, (average for all views) $\mathbf{M_m}$ is the average Model VS bin Population

$\mathbf{N_{si}}$ is the number of VS size bins, $\mathbf{N_{or}}$ is the number of VS orientation bins

(Therefore, assuming uniform distribution,

$$\mathbf{M_i = N_i / (N_{si} * N_{or})} \quad , \quad \mathbf{M_m = N_m / (N_{si} * N_{or})}\ )$$

$\mathbf{N_v}$ is the number of (θ_x, θ_y) views, $\mathbf{N_o}$ is the number of (θ_z) offset values

From our experiments, we have found the following values to be effective:

$\mathbf{N_{si}} \Rightarrow 18$, $\mathbf{N_{or}} \Rightarrow 36$ (from 10 degree quantization)

$\mathbf{N_v} \Rightarrow 178$ (15 degree step size) ; $\mathbf{N_o} \Rightarrow 24$ (15 degree quantization)

3d View Indexing

We use three different versions of 3D view indexing: 1) no orientation, 2) relative orientation and 3) absolute orientation. The efficiency of each is dependent on the amount of image clutter, complexity of the target, and resolution of Vertex Space. The first version uses size information alone, which produces many view hypotheses since size alone is much less discriminating than size and orientation together, however, the use of feature pairs, which increases the number of features to be analyzed, can be avoided. Image pairs are formed to exploit relative orientation which is encoded in the Vertex Space hash table. The resulting view hypotheses are stronger due to the combination of size and orientation discrimination, however, there are more features that must be analyzed. Finally, we encode absolute orientation into the hash table and use single features, which requires that we explicitly shift the orientation of Vertex Space to accommodate the $\mathbf{N_o}$ orientation offsets.

- No Orientation (no) Information

For no orientation information, the indexing process involves accessing the Vertex

Space Hash Table (size only) with the size value of each image vertex. At each Hash Table site there are $\mathbf{M_v^{no}}$ view entries, so there are a total of $\mathbf{M_v^{no}} * \mathbf{N_i}$ operations to be performed. $\mathbf{M_v^{no}}$ can be approximated by multiplying the probability that any given model vertex will map to a given Hash Table size bin by the number of views, which is $(\mathbf{N_m} / \mathbf{N_{si}}) * \mathbf{N_v}$ (where $(\mathbf{N_m} / \mathbf{N_{si}})$ is constrained to less than or equal to 1). So the complexity is

$$\mathbf{C1^{no}_1 = N_i * (N_m / N_{si}) * N_v}$$

- Relative Orientation (ro) Information

To use relative orientation information, the $\mathbf{N_i}$ vertices form vertex pairs with other vertices that fall within the same size bin, each of which defines a relative orientation value used to access the Vertex Space Hash Table. There are $\mathbf{N_i^2/N_{si}}$ such pairs. At each Hash Table site there are $\mathbf{M_v^{r0}}$ view entries, which result in $(\mathbf{M_v^{r0}} * \mathbf{N_i^2}) / \mathbf{N_{si}}$ operations. $\mathbf{M_v^{r0}}$, the probable number of views at a given Hash Table site, can be approximated by the probability that a given view will have at least one model vertex pair entry at a given site times the number of views. So $\mathbf{M_v^{r0}}$ is $\mathbf{N_m^2 / (N_{si}^2\, N_{or})}$ and

$$\mathbf{C1^{ro}_1 = ((N_i^2\, N_m^2) \;/ (N_{si}^3\, N_{or})) * N_v}$$

- Orientation Offset Consistency

View indexing with no orientation information or with relative orientation information defers dealing with absolute orientation values. Absolute orientation is found by determining orientation offset consistency, that is, each candidate model to image vertex match defines an orientation offset which, in turn, defines absolute orientations. For consistency, the same offset orientation must apply to all model to image vertex associations in the same hypothesis, so these groupings are all placed in the appropriate offset bin for further analysis. This operation, which follows the first indexing stage described above requires $\mathbf{N_m} * \mathbf{M_i}$ operations. This is the number of model vertices $(\mathbf{N_m})$ times the number of image vertices that could be associated with each model vertex $(\mathbf{M_i})$. Each model vertex (from a given view) is assigned to a Vertex Space bin and all image vertices that fall within the same bin are candidate matches for the model vertex. Using the uniform vertex distribution assumption, there are

$$\mathbf{M_i = N_i /\; (N_{si}\, N_{or})}$$

image vertices in a given Vertex Space bin, which we approximate with, $(\mathbf{N_i\, N_m}) \;/\; (\mathbf{N_{si}\, N_{or}})$. This term must be added to $\mathbf{C1^{no}}$ and $\mathbf{C1^{ro}}$ since it represents the second stage of the view indexing module, however, it is clearly negligible with respect to both $\mathbf{C1^{no}}$ and $\mathbf{C1^{ro}}$, so it can be ignored and we have

$$C1^{no} = (N_i\, N_m / N_{si})\, N_v$$

and

$$C1^{ro} = (\,(N_i\, N_m)^2 / (N_{si}^3\, N_{or}\,)\,)\, N_v$$

- Absolute Orientation (ao) Information

View indexing using absolute orientation information involves accessing the Vertex Space Hash Table with each of the extracted image vertices for each of $\mathbf{N_o}$ offset values. Therefore, $\mathbf{N_i * M_v^{ao} * N_o}$ operations are required, where $\mathbf{M_v^{ao}}$ is the probable number of views residing at any Hash Table element, approximated by $\mathbf{(N_m / (N_{si}\, N_{or})) * N_v}$. The complexity for view indexing using absolute orientation is

$$C1^{ao} = ((N_i\, N_m) / (N_{si}\, N_{or}))\, N_o\, N_v$$

3D View Indexing Comparison

Since the resolution of orientation bins and orientation offset bins must be similar for consistency (although they do not need to be identical), $\mathbf{N_{or} \sim N_o}$, so $\mathbf{C1^{ao} \sim C1^{no}}$.

Therefore, in terms of computational cost, indexing with no orientation information is comparable to indexing with absolute orientation information since the reduction in data per bin is directly countered by an explicit orientation offset overhead. The complexity (from above) is $\mathbf{(N_i\, N_m / N_{si})\, N_v}$, so data reduction results from subdividing into size bins, the orientation grouping does not enter into the complexity of the first search module. We compare this complexity to that of indexing using relative orientation information

$$\frac{C1^{ro}}{C1^{ao}} = \frac{((N_i N_m)^2/(N_{si}^3 N_{or}))\cdot N_v}{((N_i N_m)/N_{si})\cdot N_v}$$

$$\frac{C1^{ro}}{C1^{ao}} = \frac{N_i N_m}{N_{si}^2 N_{or}}$$

Therefore, absolute orientation indexing becomes more efficient when $N_i N_m > N_{si}^2 N_{or}$, which is the case for high data environments (complex models and high clutter, unsegmented imagery). Using the nominal values for $\mathbf{N_{si}}$ and $\mathbf{N_{or}}$ (18 and 36 respectively), $N_{si}^2 N_{or} \Rightarrow 10{,}000$. So when $\mathbf{N_i\, N_m} > 10000$, that is any combination of the number of image vertices in a group times the number of model vertices per view is greater than about 10000, absolute orientation indexing will be more efficient than relative orientation indexing.

3D View Verification and 2D Transform Determination

This process is dominated by performing all pairwise associations on the

$$\mathbf{N_c = N_m\, M_i}$$

$$\mathbf{= (N_m N_i)\ /\ (N_{si}\, N_{or})}$$

candidate image vertices for each view hypothesis. Since for N elements there are (N * (N-1) / 2) pairs, the complexity is given by $\mathbf{(N_m M_i)^2}$, so

$$\mathbf{C2 = (N_m N_i / N_{si}\, N_{or})^2}$$

Here we can see explicitly the role of Vertex Space in reducing complexity. Without Vertex Space, we would have complexity $\mathbf{(N_m N_i)}$, so Vertex Space reduces $\mathbf{N_m}$ and $\mathbf{N_i}$ each by a factor of $\mathbf{(1\ /\ N_{si}\, N_{or})}$, so the reduction in complexity is $\mathbf{(1\ /\ N_{si}\, N_{or})^2}$.

End-to-end Feature Matching

For low clutter (lc) environments where a clear peak for indexing into the view parameters (θ_y, θ_x, θ_z) can be found, only a single view need be investigated. In this case, the end-to-end complexity is just

$$\mathbf{C^{lc} = C1^{ro} + C2}$$

$$= (\ (N_i\, N_m)^2 / (N_{si}^3\, N_{or}\ )\ )\ N_v + (N_i N_m / N_{si}\, N_{or})^2$$

At the other extreme, very high clutter (hc) environments with respect to model complexity, useful peaks will be significantly degraded. In the worst case analysis, we assume that *all* views will need to be investigated. The resulting complexity is

$$\mathbf{C^{hc} = C1^{ao} + C2}$$

$$= ((N_i\, N_m) / (N_{si}\, N_{or}))\ N_o\, N_v + (N_i N_m / N_{si}\, N_{or})^2\ N_o\, N_v$$

$$= [\ ((N_i\, N_m) / (N_{si}\, N_{or})) + (N_i N_m / N_{si}\, N_{or})^2\ ]\ \ N_o\, N_v$$

where there are $\mathbf{N_v N_o}$ view hypotheses which could be investigated. For this case, the second term clearly dominates, so we have

$$\mathbf{C^{hc}} = N_o N_v (N_i N_m / N_{si}\, N_{or})^2$$

that is, the search is proportional to the number of views in the model database times $\mathbf{(N_i N_m)^2}$ reduced by a $\mathbf{(1/N_{si}\, N_{or})^2}$ factor, which acts to counter the effect of the multiple views. Essentially, we have a second order (in number of feature pairs) computation versus the direct model to image feature matching approach which is a third order problem. For large numbers of features, the computational savings is significant. The currently implemented approach does not assume any segmentation. That is, all image vertices are considered together as one large group. If grouping schemes are implemented (e.g., vertices grouped by proximity, perceptual significance,...), then only subsets of the image vertices will be considered at one time, which can greatly reduce the computational burden.

After the feature based search is performed the final phase of the search involves mapping the entire target model into the image for image domain correspondence determination. While this is a relatively costly process, it is rarely required due to the powerful prescreening ability of the feature based search.

STABLE FEATURE EXTRACTION

Feature based matching for ATR is critically dependent on the stability of extracted features. It is crucial that small changes in viewing conditions (e.g. geometry, lighting, occlusion,...) do not result in substantial changes in the features extracted. In developing our high level feature matching search strategy, the emphasis was on search techniques and the task of extracting features from the image was deemphasized. During that stage of the program, only idealized vertex features were used. Simple, idealized, wire frame, polyhedral models were used which yielded unambiguous vertices that were simple to extract. The idealized image vertices were just instances of straight line edge intersections. To extract these vertices we used straightforward standard techniques. Edges were extracted from images of simple models by calculating the gray level gradient magnitude and then thresholds. Since wire-frame models were used for rendering images, edge thicknesses were normally one pixel.

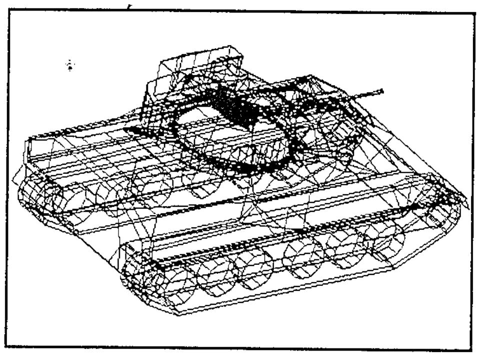
Edges were mapped to contour structures by sequentially storing pixels along the edge, which imposed pixel ordering on the extracted edges. Curvature was calculated along the contours and thresholded to find vertex sites. Curvature calculations required finding derivatives numerically along the extracted contours, which was implemented by convolving with derivatives of the Gaussian. Convolution with Gaussians has been shown to be a stable, efficient and scale selectable technique for finding derivatives of image contours.

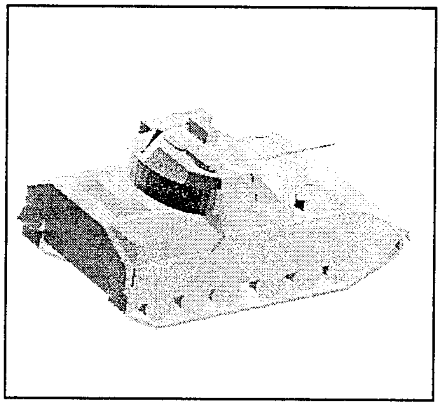
Our simple extraction techniques were adequate for development and evaluation of our search strategy, however, much more sophistication in feature extraction is required for real, complex models and imagery even though the search techniques still apply. Realistic treatment of complex models (usually containing thousands of facets) requires image rendering using artificial light sources, simple wire frame renderings are not adequate. Consequently, variations in contrast must be dealt with. Further, the complex

Fig. 14) COMPLEX 3D TARGETS - rendering and feature extraction

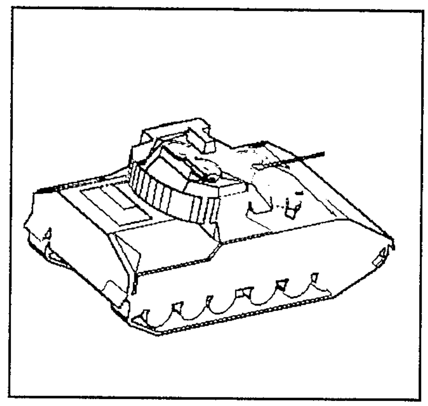
M2 Tank **(FRED file)**
1790 Facets

Wire Frame Rendering
(no hidden line removal)





Rendering with Lighting



Processed Traced Contours and Vertices

Edge Extraction from Rendered Image

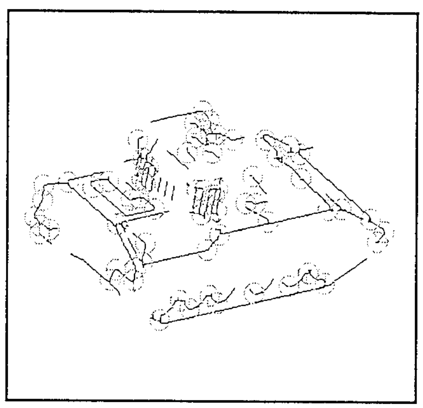
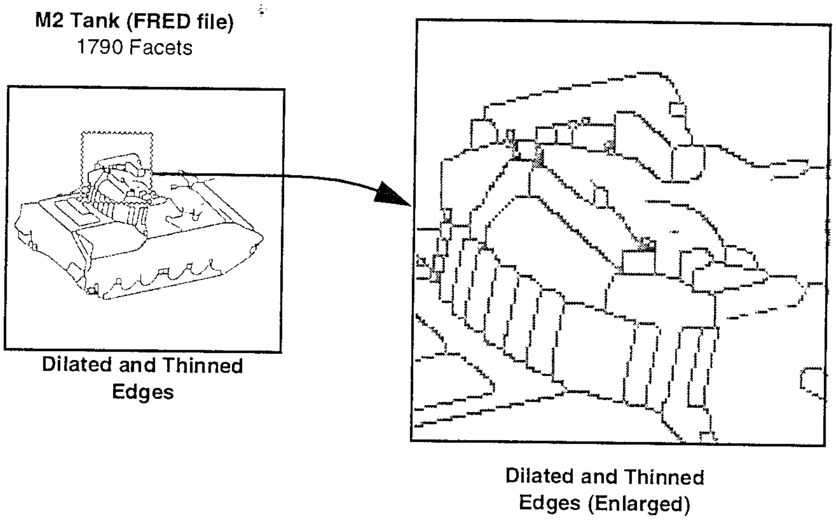


Fig. 15) COMPLEX 3D TARGETS - difficulty of curve tracing

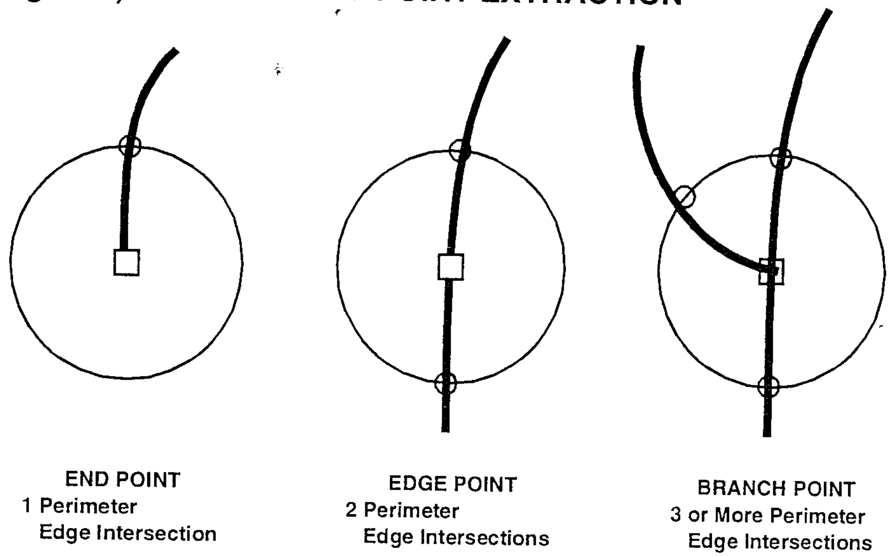


nature of the targets results in complicated topologies making contour tracing difficult and many important features, such as curvature extrema, can not effectively be treated as ideal vertices. Also, extracted edges often vary in width so that both edge breaks and thick edges are common occurrences. A more subtle problem related to our ATR application is the need to extract only significant features that aid the recognition process and reject features that can not be reliably extracted and only confuse the ATR system. A complex target model consisting of thousands of facets can easily generate so many features that the search component could be overwhelmed. So one of the requirements of the feature extraction component is reduction of target signature to manageable levels.

To address these issues, we extended the feature extraction techniques developed for simple wire frame models. To deal with broken edges and thick edges we performed morphological processing on the extracted edge image. To fill holes we performed morphological dilation, which fills in small holes and thickens edges. The thick edges cause a problem when contours are traced, since the contours can wander within the thickened edges. Therefore, we followed dilation with thinning, which reduces the thickened edges to a nominal one pixel thickness (see Fig. 14 and 15).

To deal with complex topology, which results in ambiguous curve tracing when branches are encountered (see Fig. 15), all branch and end points are identified in the processed edge image (see Fig. 16a). Then the image is scanned until a branch or end point is encountered and all connecting contours are scanned until another branch or end point is found. The basic technique used to extract branch and end points is examination of all edge points using a circle centered on the point of interest. The circle boundary is scanned to determine the number of edges encountered (Fig. 16a). One edge indicates the presence of an end point, two edges indicates a simple edge point and three or more encountered edges indicate a branch point.

Fig. 16a) BRANCH / END POINT EXTRACTION



Although the thinning process normally results in single pixel edges, there are exceptions, especially in the vicinity of branches. Therefore, non-edge to edge transitions are counted instead of edge pixels. Also, edges that approach each other to within a distance less than the circle radius used should not be considered in the edge count since the edges are not explicitly connected (small holes have already been filled by the dilation process). Therefore, we require edge pixels to be connected to the center pixel of the circle through an edge segment to be considered. This process results in branch or end point regions, which consist of groups of connected pixels at branch or end locations.

Fig. 16b) BRANCH / END POINT PROCESSING
Replace extracted branch/end region with single point surrounded by the most neighbors

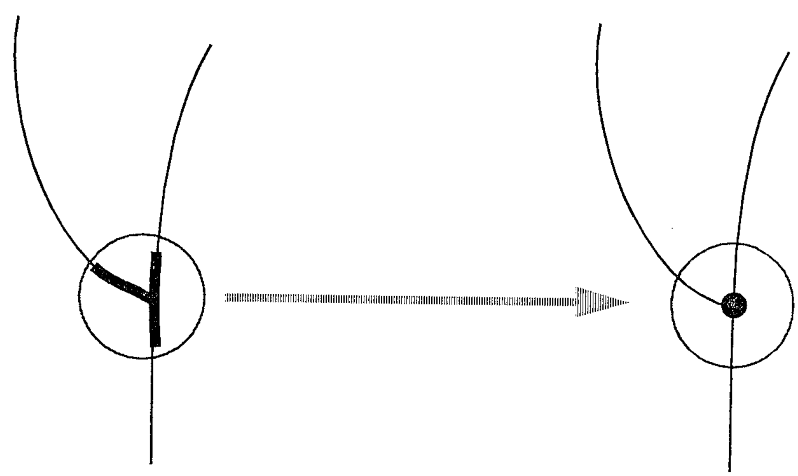
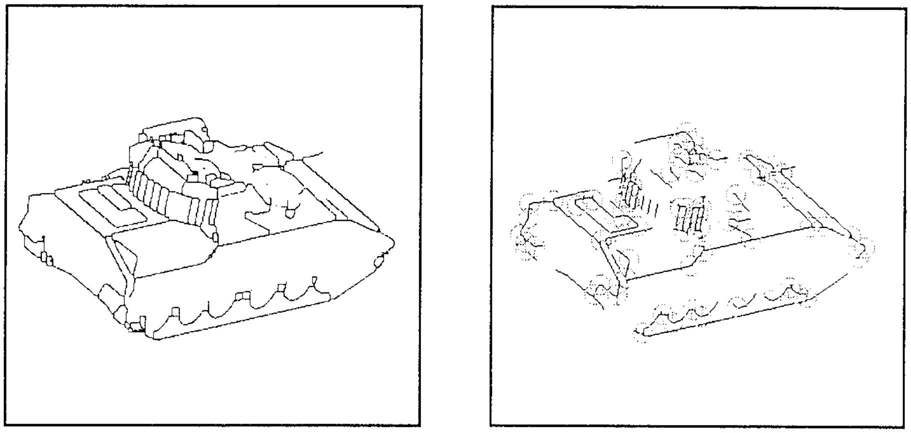


Fig. 17) COMPLEX 3D TARGETS - baseline feature extraction results



Dilated and Thinned Edges

Processed Traced Contours and Vertices

Each branch/end region is collapsed to a single pixel by replacing the region with the single branch/end point pixel with the most neighbors. This typically results in a subjectively optimum selection of branch/end points (Fig. 16b).

Any extracted contours less than a minimum length are rejected because it is likely they are not significant features that can be reliably extracted and will confuse the recognition process rather than aid it. The remaining contours are examined for curvature extrema (as described above). Vertices consist of the union of branch points and curvature extrema. A vertex thinning process, where groups of vertices within a small neighborhood are replaced by a single vertex as determined by a neighborhood filter (as described above), is used to eliminate redundant vertex detections. Vertex parameters are found by calculating angles between lines from the vertex location to edge intersections with a circle centered on the vertex site.

This process results in what appear to be reasonable vertices (Fig. 17), however, when integrated into the ATR system, recognition performance is poor. Upon examination, it can be seen that the extracted vertices are very unstable with respect to small changes in viewing geometry due largely to artifacts introduced by the thinning operation (Fig. 15). We have developed one solution, which avoids thinning by optimal contour tracing of thick edges. A filter is used that stores at each pixel the number of edge pixels in its neighborhood. Contour tracing proceeds by progressing from one filter maximum to the next a set number of pixels away. If the filter output is considered to be a 3rd dimension of the dilated edge image, then our optimum contour following is analogous to following the ridge defined by the filter output local maxima.

While this approach seems promising and may provide a useful alternative to morphological thinning, we believe that vertex features can best be extracted using a direct local approach that does not require explicit edge and contour extraction. The topology of complex objects can be arbitrarily complicated, making any explicit edge following technique difficult and limited in applicability. Further, direct treatment of scale is required since scale can have a significant effect on the nature of vertex features extracted from imagery. In addition, calculated vertex parameters should be stable with respect to the precise location of an extacted vertex.

To address these issues, we consider the geometric definition of curvature to motivate our approach.

If

F(x,y) is the image gray level at pixel site (x,y) and

C(x,y) is a contour pixel where the gradient of F(x,y) is a local maximum and

T(x,y) is the UNIT tangent of C at (x,y) and

N(x,y) is the UNIT normal to C at (x,y) (gradient direction)

so $\vec{N} = \left(\frac{F_x}{Q}, \frac{F_y}{Q}\right)$ **where** $Q = \sqrt{F_x^2 + F_y^2}$ **, the magnitude gradient**

then Curvature is $K(x, y) = \left|\frac{d}{ds}\vec{T}(x, y)\right|$ **where** $\frac{d^\circ}{ds}$ **is the derivative of** $^\circ$

along the curve contour in the tangent direction

let $\vec{R} = \frac{d}{ds}\vec{T}(x, y)$ **and the components of** $\vec{R}$ **be Rx and Ry,**

such that $\vec{R} = (Rx, Ry)$ **, then** $K = |\vec{R}|$.

So $Rx = \left(\frac{d}{dx}Tx \cdot Tx\right) + \left(\frac{d}{dy}Tx \cdot Ty\right)$ **and** $Ry = \left(\frac{d}{dx}Ty \cdot Tx\right) + \left(\frac{d}{dy}Ty \cdot Ty\right)$

we know from the properties of the unit tangent and unit gradient that

$\vec{T} \bullet \vec{N} = 0$ **(from orthogonality); and** $(Tx)^2 + (Ty)^2 = 1$

so $(Tx \cdot Nx) + (Ty \cdot Ny) = 0$

therefore $Tx = -\left(\frac{Ny}{Nx}\right) \cdot Ty$ **and** $(Tx)^2 = \left(\frac{Ny}{Nx}\right)^2 \cdot (Ty)^2$

Then (with $\frac{dF}{d^\circ} = F_\circ$**)**

$$K = \frac{([(\mathbf{F}_{yx} \cdot \mathbf{F}_y) - (\mathbf{F}_{yy} \cdot \mathbf{F}_x)]^2 + [(\mathbf{F}_{xy} \cdot \mathbf{F}_x) - (\mathbf{F}_{xx} \cdot \mathbf{F}_y)]^2)^{1/2}}{\mathbf{Q}^2}$$

(where $\mathbf{Q} = \sqrt{\mathbf{F}_x^2 + \mathbf{F}_y^2}$**)**

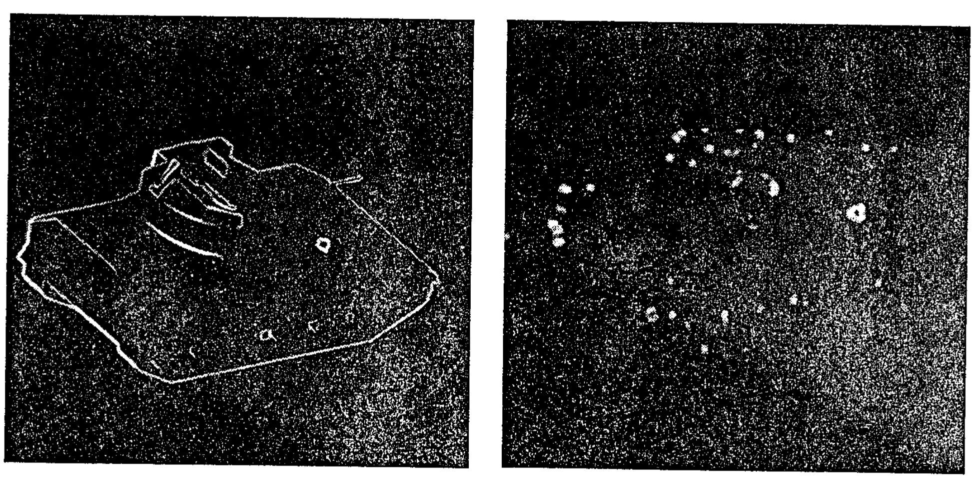
and finally, after noting that F_{xy} and F_{yx} are equivalent, we have

$$\boxed{K = \frac{([(\mathbf{F}_{xy} \cdot \mathbf{F}_y) - (\mathbf{F}_{yy} \cdot \mathbf{F}_x)]^2 + [(\mathbf{F}_{xy} \cdot \mathbf{F}_x) - (\mathbf{F}_{xx} \cdot \mathbf{F}_y)]^2)^{1/2}}{\mathbf{F}_x^2 + \mathbf{F}_y^2}}$$

While this expression for curvature has a theoretical basis, there are many practical issues that must be considered. One issue is the well known numerical instability of calculating high order spatial derivatives of images. Numerical differentiation tends to accentuate high frequency noise and the effect is more pronounced for successively higher orders of differentiation. Also, in our application, we are most interested in small contour regions where branching or a high level of curvature occurs. These are regions where good numerical differences, which are required for the derivative calculations, are not well defined because image gray scale is changing quickly and in complex ways. There remain the questions of scale analysis and stable vertex parameter determination.

Therefore, although we have implemented the curvature calculation derived above (Fig. 18), we use it primarily as a point of departure. Rather than using numerical

Fig. 18) DIRECT CURVATURE CALCULATION
Using Principal Axes for Orientation



Gradient Magnitude of Tank Image

Extracted Curvature

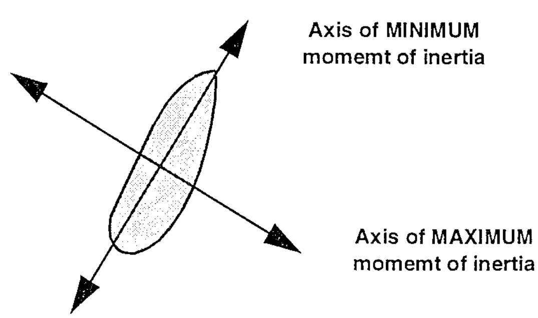
curvature explicitly, we use it as a conceptual guide to a more practical implementation. Conceptually, geometric curvature is a measure of the change in orientation along an edge contour. Using this curvature concept, we proceed by finding local orientation at all potentially interesting points (points where there is a non-zero gradient magnitude) in the image. The calculated orientation is analyzed by finding points where local orientations implicitly intersect. This is achieved by implicitly projecting a small distance in the direction of the local orientation and checking for the existence of a different orientation along the projection. For stable vertex detection, the implicit orientation change conditions described here must persist over some specified scale. Therefore, points of high curvature are found not only based on absolute changes in orientation but also by requiring the changes to occur along the direction of orientation, which is consistent with the spirit of geometric curvature.

To determine orientation at the lowest scale (highest sensitivity), we use the components of the gradient, where the x and y difference components are found using 1x3 (and 3x1) kernels. However, sensitivity is achieved at the cost of stability. To get more stable values at higher scale we would normally increase the size of the kernels used to calculate the derivatives, however, this often results in increasing the ambiguity of the gradient calculation. To see one example of this, consider a strong, but narrow vertical step edge in an image (from left to right, light to dark to light). For a small kernel, the gray level difference in the x direction will be maximum at the optimal location where the gradient kernel straddles the step edge. If we increase the size of the kernel beyond the end of the step edge the x component value will decrease and the gradient output will be correspondingly degraded.

Alternatively, we calculate the principal axes of a region centered about each point of

Fig. 19) IMPLICIT EDGE ORIENTATION DETERMINATION

- **PRINCIPAL AXIS DIRECTION**
  (high stability, large kernel size)
  - principal axes are associated with maximum and minimum moments of inertia, and align with predominant directions



interest. Principal axes define dominant directions, or orientations in a region. In rigid body analysis, principal axes define axes of stable rotation for an object and will necessarily coincide with any axes of symmetry (see Fig. 19). To apply the principal axis calculation to imagery, mass density becomes the image gray scale value and we use only the two image dimensions, x and y. Principal axes are found by solving the two-dimensional Eigenvector matrix equation $\mathbf{I}\vec{\mathbf{R}} = \lambda\vec{\mathbf{R}}$ where the two Eigenvector solutions are the principal axes and are necessarily orthogonal to each other. Here I is the moment of inertia matrix whose elements are

$$\underset{\sim}{\mathbf{I}} = \begin{bmatrix} \mathbf{I}_{xx} & \mathbf{I}_{xy} \\ \mathbf{I}_{xy} & \mathbf{I}_{yy} \end{bmatrix}$$

and $\mathbf{I}_{jk} = \sum_{\mathbf{region}} [\mathbf{F}(\mathbf{x}, \mathbf{y})(\mathbf{r}^2\delta_{jk} - \mathbf{x}_j\mathbf{x}_k)]$ where $\mathbf{F}(\mathbf{x}, \mathbf{y})$ is the image gray level at location (x,y) and $\mathbf{r}^2 = \mathbf{x}^2 + \mathbf{y}^2$. The matrix equation yields a secular equation, $|\mathbf{I} - \lambda\Lambda| = 0$ where Λ is the identity matrix. So the Eigenvalues (moments of inertia about each axis) are

$$\lambda = \frac{(\mathbf{I}_{xx} + \mathbf{I}_{yy}) \pm \{(\mathbf{I}_{xx} + \mathbf{I}_{yy})^2 - 4(\mathbf{I}_{xx}\mathbf{I}_{yy} - \mathbf{I}_{xy}^2)\}^{1/2}}{2}$$

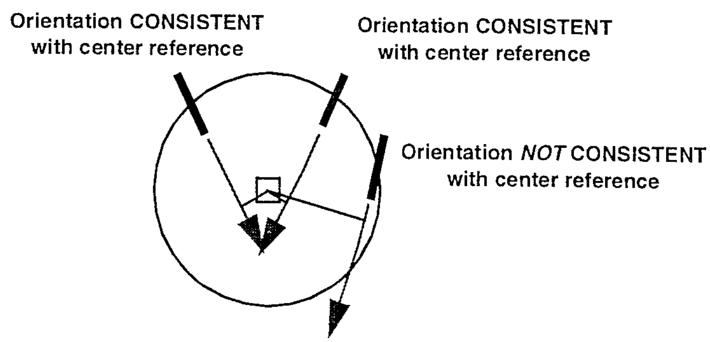
and $\mathbf{I}_{xx}\mathbf{Rx} + \mathbf{I}_{xy}\mathbf{Ry} = \lambda\mathbf{Rx}$ or $\mathbf{Rx} = \left(\frac{\mathbf{I}_{xy}}{\lambda - \mathbf{I}_{xx}}\right)\mathbf{Ry}$, which defines the relations between the Eigenvector components. To get actual component values, we impose the constraint that the Eigenvectors be unit length ($(\mathbf{Rx})^2 + (\mathbf{Ry})^2 = 1$). This procedure yields two Eigenvectors, one parallel to the orientation and one perpendicular. We choose the vector that corresponds to the minimum gray scale variance, which should be parallel to the local orientation A combination of the gray scale variance and ratio of the two Eigenvalues serves as a quality metric for each orientation value (small variances and large Eigenvalue ratios correspond to good, unambiguous orientation values).

Having found the local orientations, we often wish to do some postprocessing, such as weighted average smoothing using orientation quality measures for the weighting. For any manipulations of this kind it is important to remove inherent orientation discontinuites that result from the discontinuous numerical values used to describe continuous orientations (e.g., an orientation of 1 degree is relatively close to a 359 degree orientation although the numerical difference is large).

First, we note that, for our purposes, the direction sign of a vector is superfluous since

Fig. 20) IMPLICIT EDGE ORIENTATION ANALYSIS - CRITICAL POINT DETECTION

- **Find consistent surrounding orientations**
- **Compare consistent orientation values**



the same orientation is described equivalently by a vector and its negative. To remove this redundancy, we impose the constraint that all vectors be normalized to point to the first or second quadrant (0 to 180 degrees), which can be implemented by negating any vector whose y component is less than 0. The orientation angle of the vector is then doubled so the range is 0 degrees to 360 degrees. While the orientation values are not continuous, the corresponding transformed vector components are. Therefore, all operations are performed separately on the transformed vector components since vectors that define similar orientations will necessarily have similar transformed vector components and numerical discontinuites will be avoided. After processing, the inverse operations are performed on the resultant vector to find the true orientation.

The processed image orientation field is analyzed to extract generalized vertices which do not necessarily fit the idealized vertex model of two intersecting straight lines. A small circle centered on each point of interest (any point not inside a uniform gray level region) is used to examine the neighborhood. Any point on the circle whose orientation projects near enough to the center reference point is considered to be on the same implicit contour as the reference point, regardless of other region details (see Fig. 20). Therefore, two points can be assigned to the same implicit contour even if there exist contour "holes" between the points. If multiple adjacent points on the circle are found to be implicitly connected to the center reference point, they are replaced with the single pixel whose orientation projects closest to the center point, which effectively insures that implicit contours will be one pixel thick.

This process leaves one pixel on the circle for each implicit contour segment. These pixels are further examined for orientation consistency by stepping outwards in the direction of their orientation. If the priority is sensitivity, it is sufficient that the orientation at each step be consistent (project near enough) with the previous pixel. If, however, stability is required over sensitivity, the orientation at each step must be consistent with the original reference pixel, which is a more restrictive condition than that used to achieve higher sensitivity. In the former case, curved contours can be extracted while the latter case requires that contours be essentially straight within the range of consideration. By varying the range of consideration we can control the scale and stability of extracted vertices so that very small features will not be extracted even if they appear to be highly curved.

Implicit contour segments that are orientationaly consistent over the required steps are used to determine whether or not a vertex is present. More than two implicit contour segments indicates a branch point and, therefore, also a vertex. If there are identically two contour segments, the difference in orientation at their ends is used as a measure of curvature. If the curvature is greater than the necessary threshold, a vertex is extracted and the calculated curvature value is used for the vertex size (Fig. 20). This approach results in stable vertex angle values with respect to small changes in the location of the vertex since absolute locations are not used to determine angles, only consistent orientation values in the neighborhood of the vertex location are used.

This vertex extraction process will typically result in groups of vertex pixels clustered together. For a stable vertex, the conditions that determine vertex existence must exist over a small region so there will necessarily be multiple vertex pixels output from the vertex extraction process. The raw vertex pixels (vertex locations) are processed so as to replace localized groups of pixels with a single optimum vertex pixel. First, a size filter is used that filters out all candidate vertex sites that are not part of a connected region of neighbor vertex sites greater than a minimum size threshold. The resulting vertex sites are dilated (thickened) so that close vertices are merged. The final processing stage collapses vertex site regions down to a single vertex pixel by replacing the region with a single "best" pixel based on curvature and gradient strength.

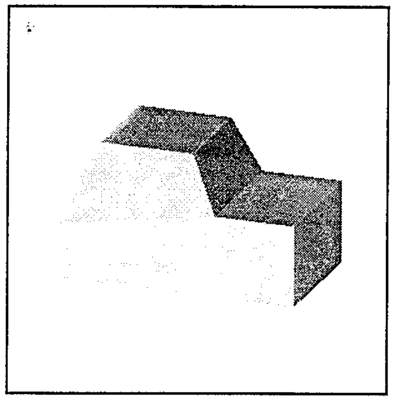
PREVIOUS RESULTS

Search Strategy Results

To test and evaluate our MBATR search strategy while de-emphasizing the role played by feature extraction, we have used synthetic imagery and rudimentary feature extraction techniques not appropriate for more complex imagery. We have generated simple 2D and 3D models and implemented interactive tools for manipulating and rendering images of the generated models.

One of the primary issues is ATR performance in the presence of clutter. To help analyze clutter effects, we can interactively add synthetic views of different objects to an image of an object we wish to recognize. To systematically obtain more analytic clutter performance data, we randomly add clutter vertices to an image of a target to be

Fig. 21) SIMPLE 3D TRUCK MODEL



recognized, in some cases with the target view unobstructed and clearly visible, in others, we obscure much of the target view and create large breaks in the contour of the object.

As one example, we have used a simple, idealized truck for a 3D target (see Fig. 21). We then generated a representative synthetic view of the truck, extracted the features, obscured many of the extracted features and added random clutter vertices to the image (see Fig. 22). Our ATR algorithm was used to correctly identify candidate matching features (see Fig. 23). To further investigate performance in clutter, we systematically added clutter features one at a time to a portion of a target signature, measured the

Fig. 22) SIMPLE 3D TRUCK MODEL MATCHING:
INPUT IMAGE WITH 394 CLUTTER VERTICES

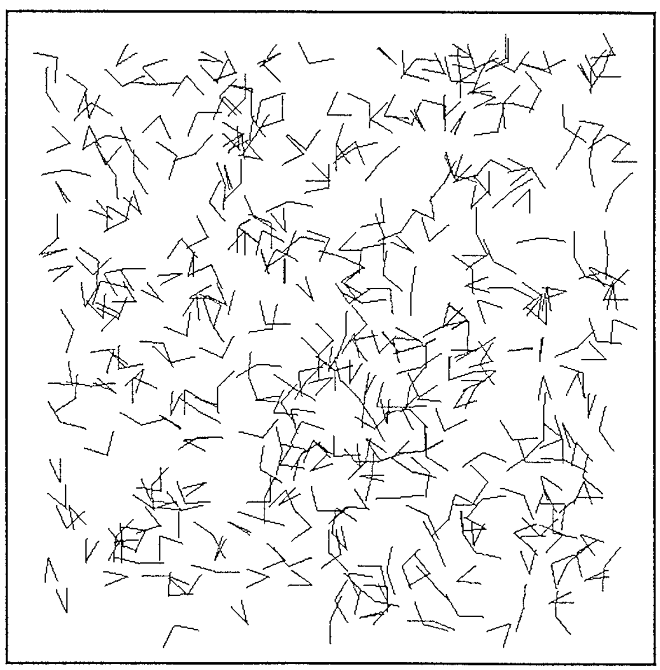
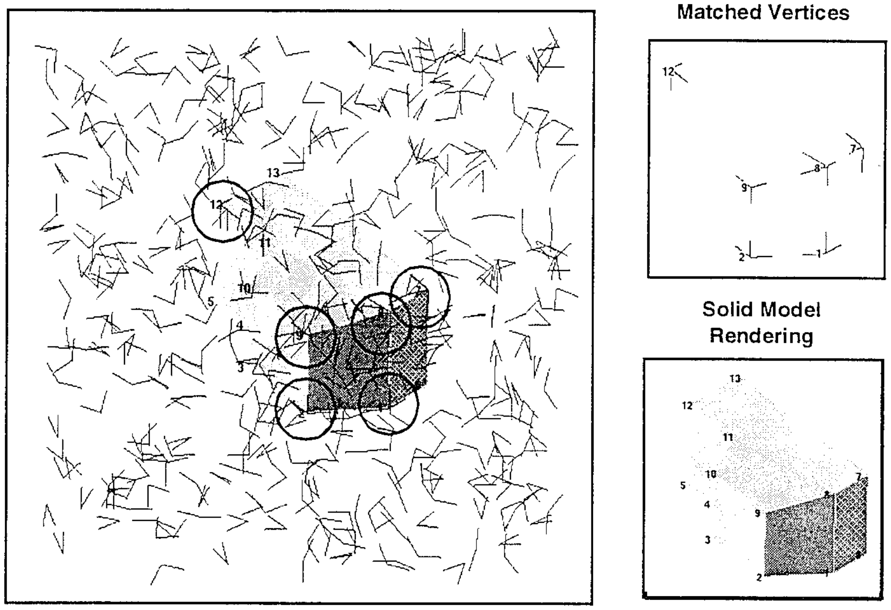


Fig. 23) SIMPLE 3D TRUCK MODEL MATCHING: RESULTS and GROUND TRUTH



recognition elapsed time and stored the result in a file so that we can quantitatively study the relation between clutter and recognition time (see Fig. 24).

Stable Feature Extraction for Complex Targets

Using the implicit edge orientation approach optimized for sensitivity, described above, we demonstrated preliminary feature extraction results with real FLIR imagery, as presented in "Automated Missile Aim Point Selection Technology Final Report" for ONR contract #N00014-92-C-0087. However, that work was considerably expanded in our most recent program and the results are presented below.

Recognition of Complex Targets

We used a model of an M2 tank consisting of 1790 facets to evaluate treatment of complex targets. Stable features were extracted from a typical synthetic target view and successfully used to match corresponding features with the best view stored in the target data base (see "Automated Missile Aim Point Selection Technology Final Report" for ONR contract #N00014-92-C-0087). However, we used only synthetic imagery since, at that time, we did not have target models consistent with our imagery. In our most recent program, we have acquired consistent sets of target models and IR imagery and have demonstrated feature matching with real targets and images, presented below.

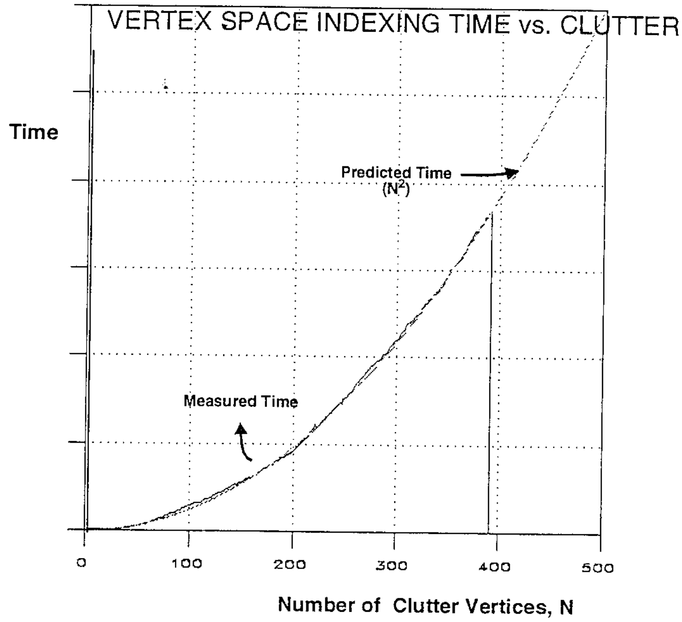


Fig. 24) VERTEX INDEXING CLUTTER TESTS - results
Indexing time versus clutter is measured by systematicaly adding random vertices to a degraded image of a target and recording the elapsed time for indexing. Note that the measured data is in close agreement with that predicted by the complexity analysis, namely, complexity should be proportionate to the square of the number of clutter (image) vertices.

RECENT PROGRESS

We acquired IR imagery databases of targets of interest from Eglin Air Force Base (TABILS), Night Vision Electro-Optical Lab (NVEOL Comanche data) and Wright Patterson Air Force Base (WPAFB). We evaluated the preliminary feature extraction techniques developed on earlier ONR programs, as described above. Our existing curvature analysis techniques proved quite effective on the IR imagery data we used for evaluation. However, we did make some minor refinements to optimize the calculated curvature. It was then clear that target recognition accuracy performance, in general, and vertex feature extraction, specifically, was limited by poor assignment of unique vertex points to curvature regions. Our vertex extraction technique generates a single vertex feature point for each region of high curvature, which is the process that required further investigation.

The problem was particularly pronounced in middle to low resolution imagery of complex targets, such as tanks, since many of the significant features are close enough together that their associated curvature regions merge into extended amorphous blobs. Our original assumption was that curvature regions would tend to be highly localized and almost any point in the region would serve well as the unique vertex location. This assumption applies to high resolution imagery, simple targets or low sensitivity feature extraction. However, when the practical ATR problem and real battlefield conditions are addressed, required performance can only be achieved by sensitive and stable feature extraction of image features with minimal spatial extent. Since, as in most ATR systems, the extracted features drive the higher level model to image matching processes, ATR performance is completely dependent on successful feature extraction. Therefore, optimum feature extraction is a goal that must be given highest priority

To extract optimal vertices from calculated curvature regions, many inter-related problems had to be addressed. The raw curvature data is noisy, and small features close together interact resulting in merged curvature regions. Further, the curvature regions are often extended so finding a single vertex point to represent the region can be a problem. And the feature attributes (such as vertex orientation and size) must not be overly sensitive to the location of the vertex. To deal effectively with these problems, we developed, implemented and demonstrated a curvature processing scheme (see Fig. 25). We start by calculating the orientation field and resulting curvature (see Figs. 25a,b,c) using the techniques described above.

As can be seen in Fig. 25c), many of the curvature regions are extended, making their locations ambiguous, and, in many places, curvature regions from different target structural features have merged together. To observe these effects in greater detail, see Fig. 25d) which displays a blown up view of an interesting sub-region, the plate at the base of an M60 tank gun barrel, of the image. Ideally, we would like to extract 4 vertices associated with the corners of the target image component (points 1-4 in Fig. 25d) even though they are only a few pixels apart. Clearly, highly sensitive and stable processing

Fig. 25) CURVATURE PROCESSING:

Fig. 25a) **Input Image:**

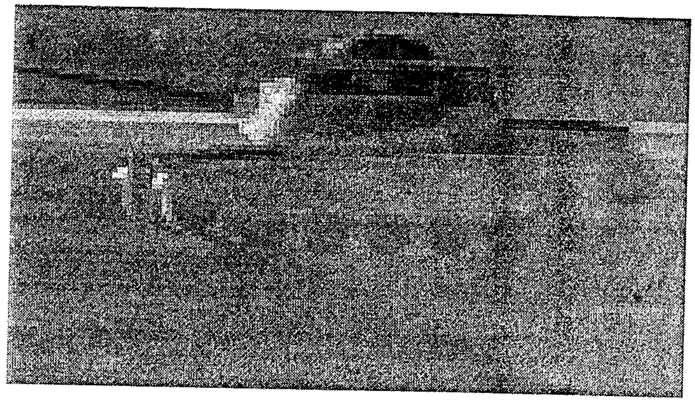


Fig. 25b) **Orientation Field:**

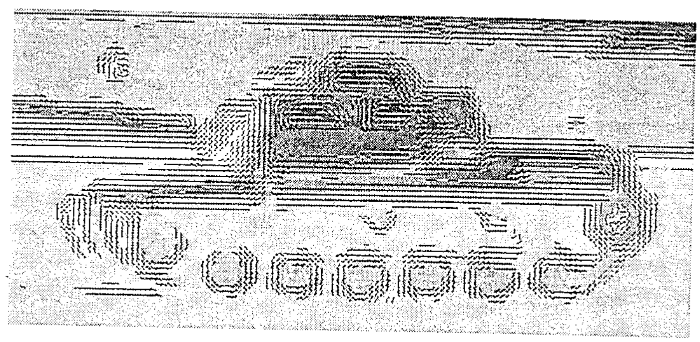
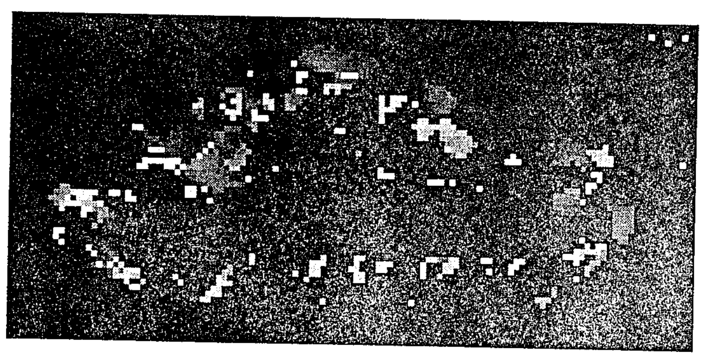


Fig. 25c) Raw **Curvature:**

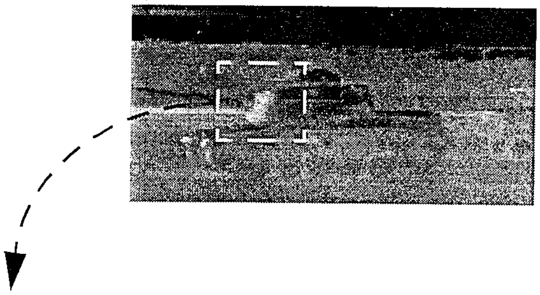


and extraction techniques are required to achieve the necessary level of precision.

First, a non-linear filtering process is performed to reject isolated curvature pixels that differ significantly from their neighbors. The nature of the curvature analysis process, described above, is such that conditions for high curvature will exist over at least a small region if a meaningful vertex is present. Therefore, isolated curvature points indicate noise and are rejected (see Fig. 25f,g). After this filtering, the curvature regions associated with the desired points V1 and V4 remain merged into one extended region. The goal is to split this region into two, one each to be associated with each intended vertex, V1 and V4.

Standard gradient thresholding techniques applied directly to the curvature data are not very effective since curvature strength does not typically vary reliably over different

Fig. 25) CURVATURE PROCESSING (continued):



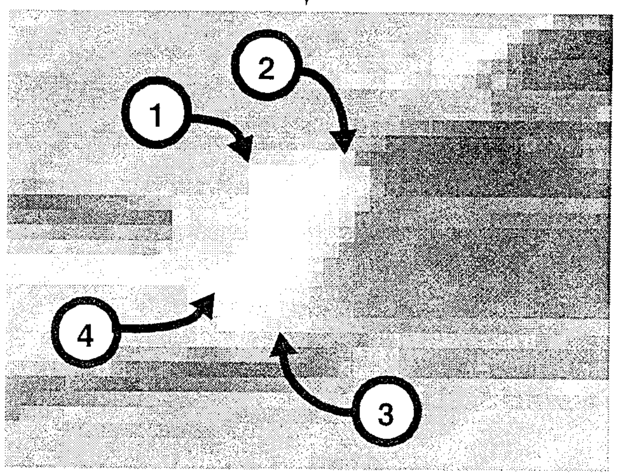


Fig. 25d) Sub-image - Histogram Enhanced
Corner Points of interest,1-4, are indicated

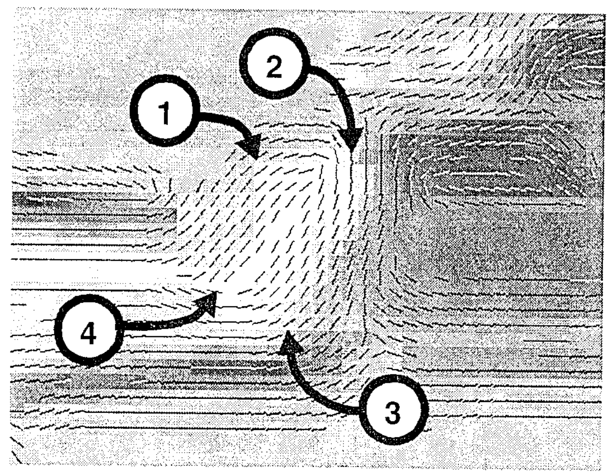


Fig. 25e) Orientation Field

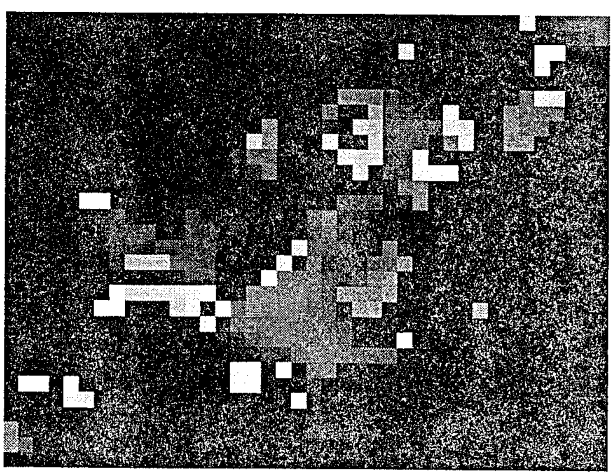


Fig. 25f) Raw Curvature

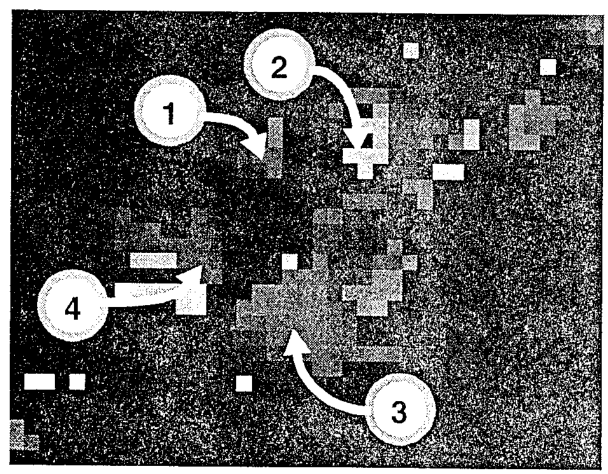


Fig. 25g) Curvature -
Non-Linear Low Pass Filtered
(1 and 4 connected by extended region)

parts of curvature regions. However, curvature orientation, properly defined, provides a much better measure for discriminating between different curvature sub-regions. Consistent with our notion of a vertex is a local vertex orientation that is a by-product of our curvature analysis and describes the orientation of the line bisecting the vertex local tangent. Curvature orientation is quite different for sub-region 1 than sub-region 4 (see Fig. 25h). Changes in both curvature orientation and strength are quantified by calculating a four dimensional gradient (Fig. 25i), which is then thresholded (Fig. 25j) and used to successfully split extended curvature regions (Fig. 25k). To deal with the sparse nature of the data, we have implemented a conditional kernel element gradient scheme. To avoid skewing gradient calculations due to partial kernel support, we only include input from a kernel element if it and its symmetric counterpart are both present in the curvature data. If so, it is processed normally, otherwise, it and its counterpart are not processed and the normalization factor is adjusted appropriately to account for the absence of the data pair. This provides for accurate gradient calculations over incomplete kernel data support in keeping with our goal of dealing effectively with limited data, as required for real battlefield conditions.

The next curvature processing step is size filtering the surviving curvature regions. The size filter metric is the number of connected pixels in each region. As mentioned above, valid vertices should have strong curvature support over at least a small curvature region, otherwise they are rejected (Fig. 25l). For the size filter we have chosen, the small surviving curvature region for vertex 2 was just below the threshold (Fig. 25k) and was consequently rejected. The reason can be seen by examining the orientation flow field (Fig. 25e.). The orientation flow support for region 2 derives primarily from two flow regions that are almost parallel (below and to the left of point 2) and separated by a small break and could well be interpreted as two segments of the same extended image edge. The extent of the horizontal flow field to the left of point 2 is very limited and results in a correspondingly limited curvature region just below the threshold we are currently using. (The effect of a small change on the size threshold can be seen below.)

The final curvature processing step assigns a vertex location to each surviving curvature region. This is accomplished by finding the number of neighbor pixels, pixels in the same region and within a given distance, at each pixel location, which provides a measure of relative thickness (Fig. 25m), that conceptually indicates how deeply each pixel is embedded in the region. The vertex location for each region is assigned to the pixel with the highest thickness value which relates roughly to the centroid of each curvature region.

Vertex parameters (size and orientation) are calculated using the orientation flow field values at the vertex end locations, so vertex location only indirectly affects the value of vertex parameters, providing improved stability over calculations using location directly.

The final extracted vertices are shown for the sub-image and the full image in Fig. 26.

Fig. 25) CURVATURE PROCESSING (continued):

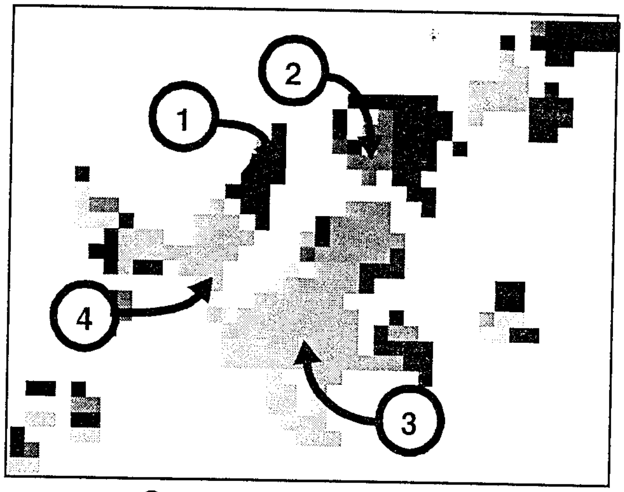


Fig. 25h) **Curvature (Vertex) Orientation**
(sub-regions 1 and 4 have different orientations)

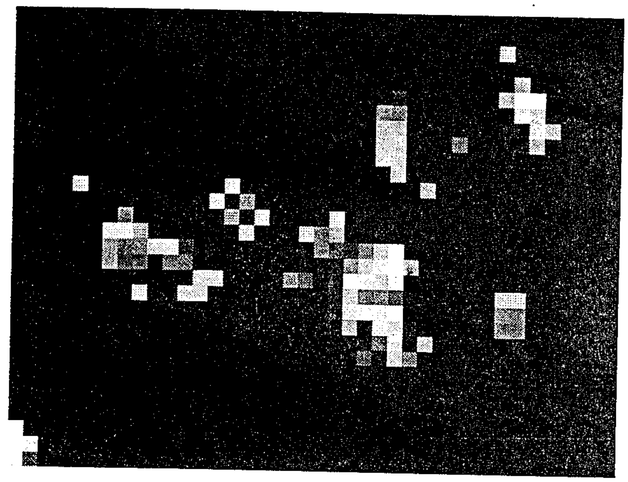


Fig. 25i) **Curvature (Strength and Orientation) Gradient Magnitude**

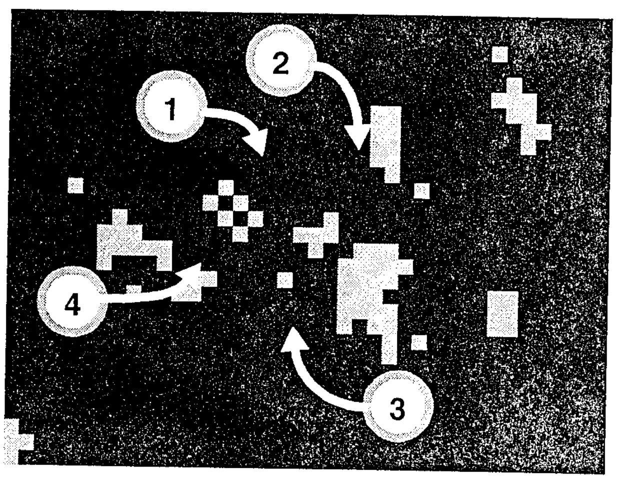


Fig. 25j) **Thresholded Curvature Gradient**

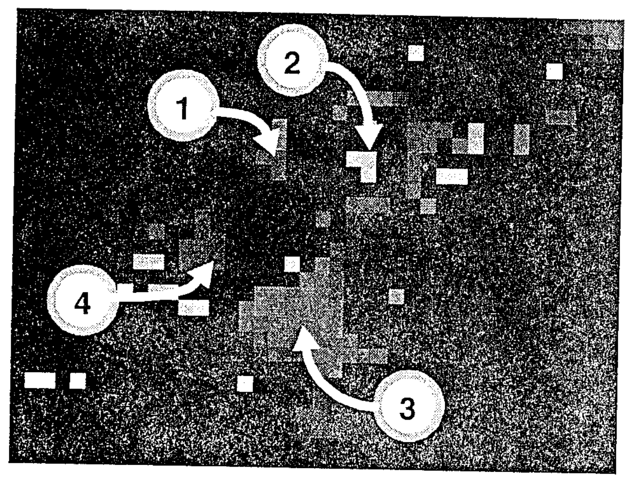


Fig. 25k) **Curvature Region Split**
(1 and 4 successfully split)

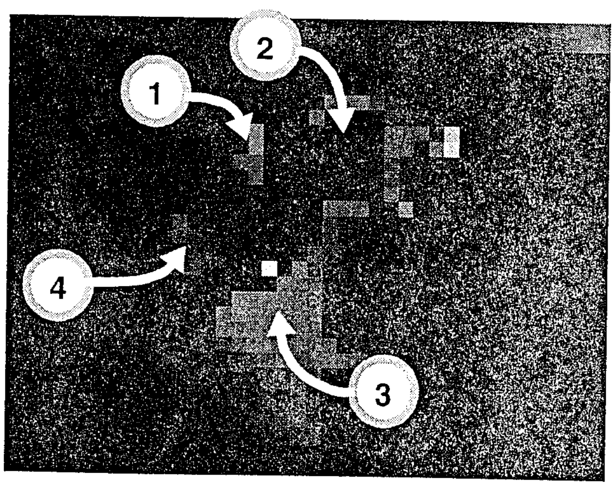


Fig. 25l) **Curvature Region Size Filtered**

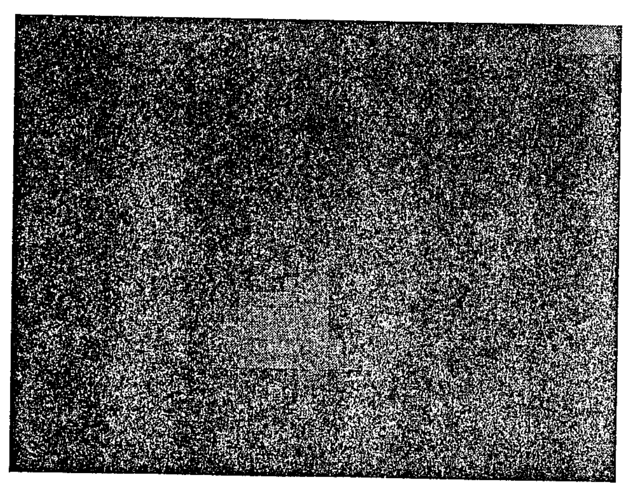


Fig. 25m) **Curvature Region Thickness**

Fig. 26) CURVATURE PROCESSING: Extracted Vertex Features

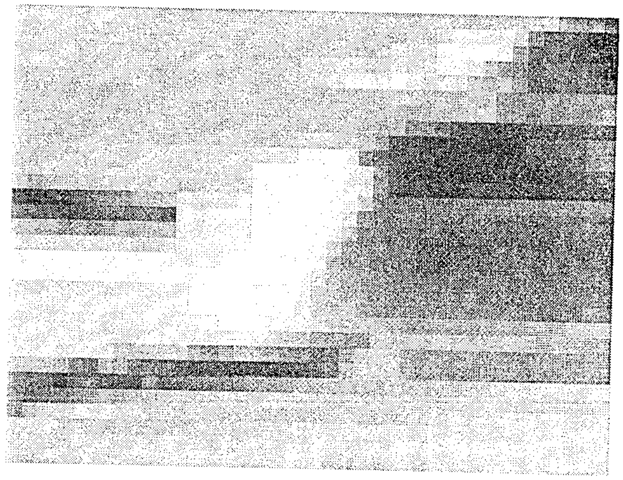


Fig. 26a) Original Sub-image

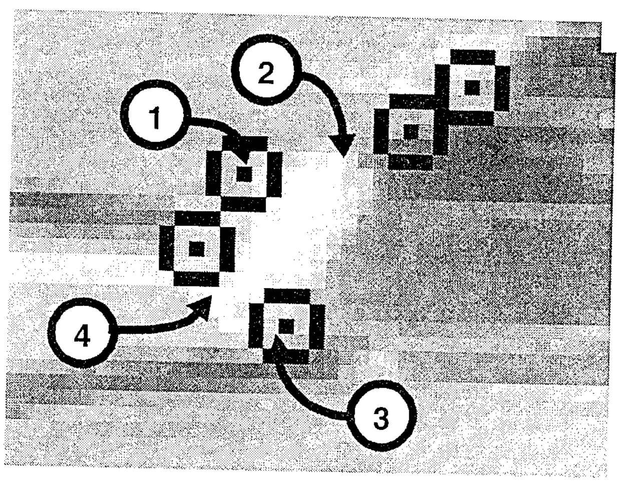


Fig. 26b) Extracted Vertices - Sub-image

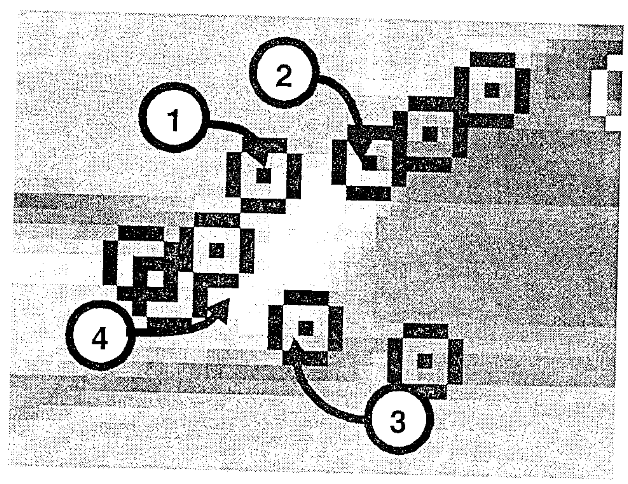


Fig. 26c) Extracted Vertices - Sub-image
Reduced Size Filter Threshold

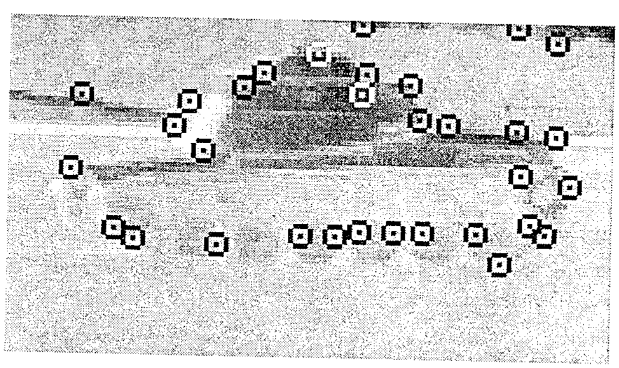


Fig. 26d) Extracted Vertices - Full Image

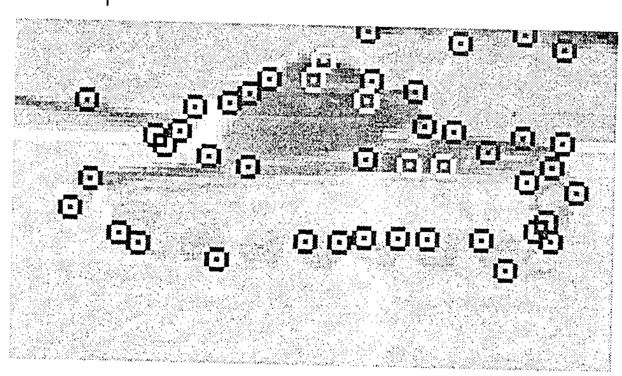
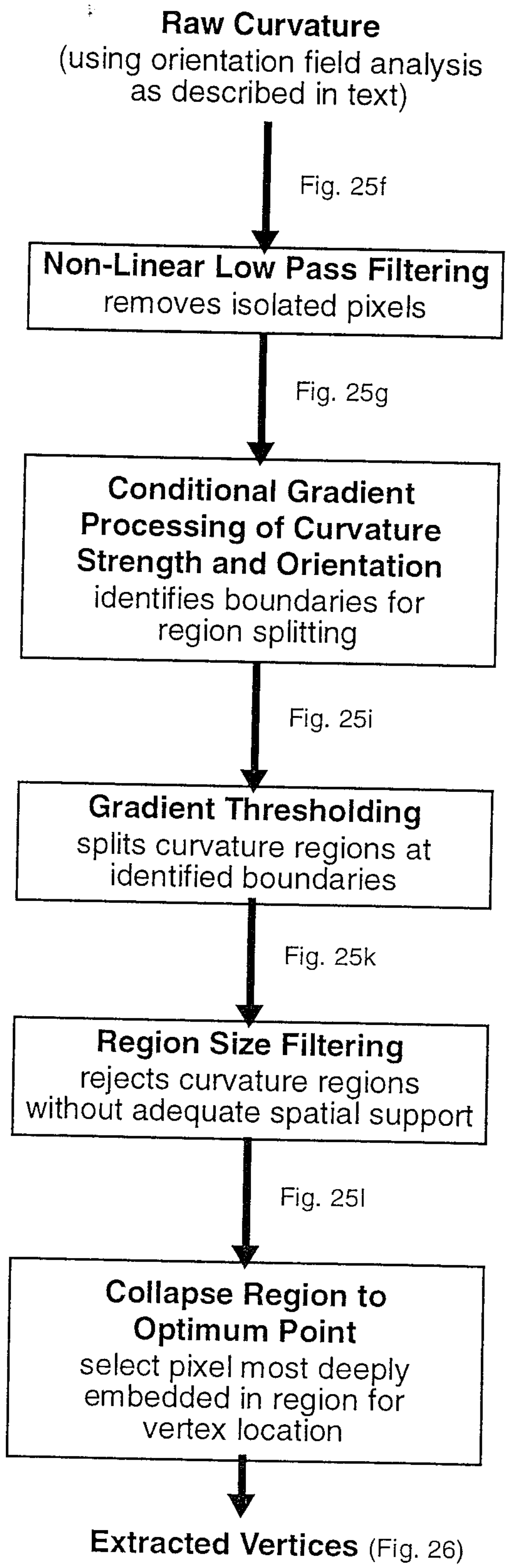


Fig. 26e) Extracted Vertices - Full Image
Reduced Size Filter Threshold

Fig. 27) FUNCTIONAL CURVATURE PROCESSING DESCRIPTION



The current and somewhat lower size threshold results are shown. The curvature processing scheme described here is presented graphically in Fig. 27.

Once the required curvature processing techniques had been developed, implemented and demonstrated, and new models were acquired, new model databases needed to be created using the new feature extraction process and new models. However, our model manipulation and database creation software was written using the Silicon Graphics Inc. (SGI) GL graphics library, which is being phased out in favor of a new multi-platform standard, OpenGL. We chose to rewrite our graphics software, using OpenGL, in part to maintain optimum support and, in part, to achieve multi-platform porting capability. Our software can now run on SGI, Suns and even Pentium Unix (Linux) platforms.

We integrated the new feature extraction and graphics software into our system and began evaluation of indexing/matching capabilities using real IR imagery and the appropriate target model. We have been using an M60 model and associated imagery. The model was provided by KRC based on BRL-CAD data.

We have demonstrated the ability to effectively match model features with real IR imagery (see Fig. 26). Although our results are recent and have not been fully evaluated, it was soon evident that there are discrepancies between the provided model geometry and the actual target geometry. Upon investigation, we were able to find at least three different instances of the M60, the A1, A2, and A3, some of which have significantly different features. We were not able to find out which versions we had for our imagery or model. It was necessary to loosen some matching thresholds accordingly to accommodate the differences. It is likely scale processing to reduce sensitivity to target feature discrepancies will also be needed to make the most effective use of the existing geometry models. These are areas that require further investigation.

Fig. 28) MATCHING REAL IR IMAGERY WITH M60 MODEL:

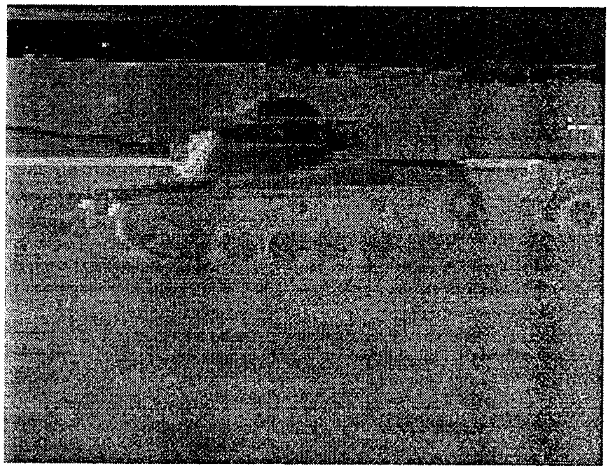


Fig. 28a) 60 Tank IR Image
original image

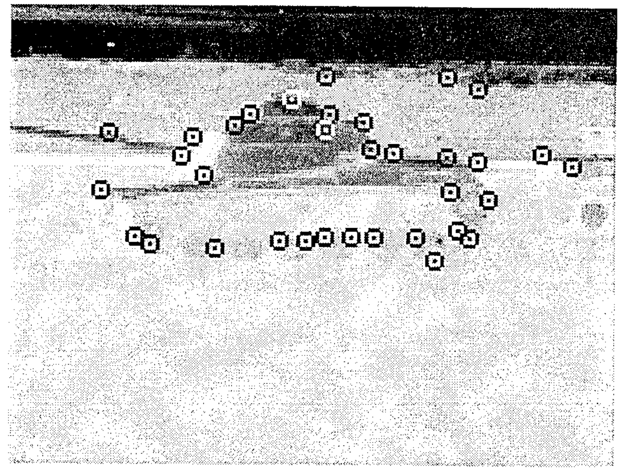


Fig. 28b) M60 Tank IR Image
extracted vertices

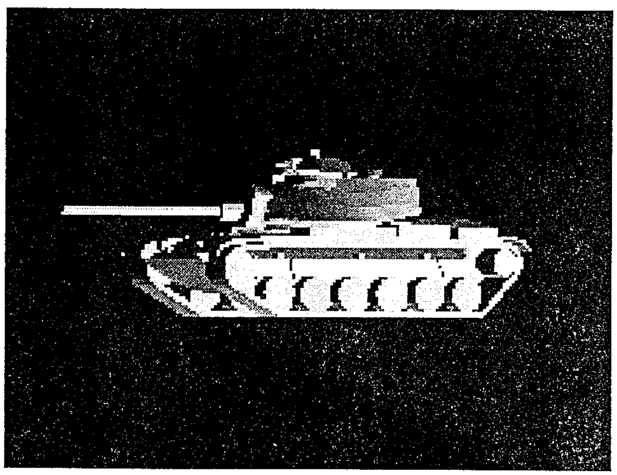


Fig. 28c) Best Matched Model View

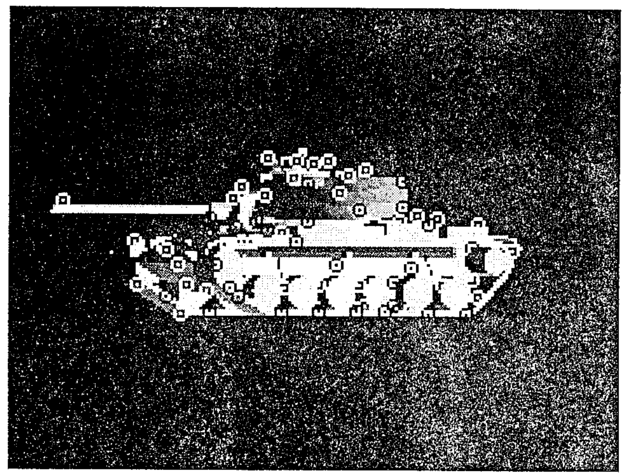


Fig. 28d) Best Matched Model View
extracted vertices

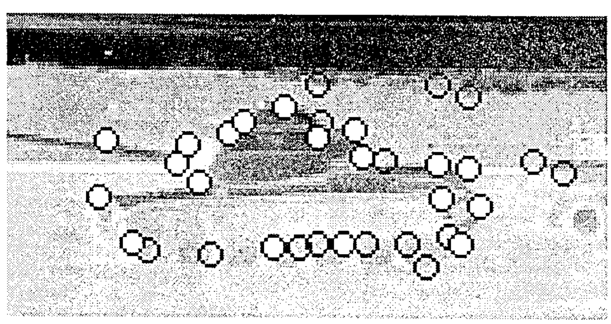


Fig. 28b) M60 Tank IR Image
vertices matched with model

PROGRAM SUMMARY

To date, we have developed powerful ATR technology that effectively addresses the critical issues driving the implementation and fielding of practical ATR systems. Inherent in our ATR approach is a treatment of partially occluded, degraded target signatures, clutter and sensitivity allowing for effective use of available information content to distinguish similar targets. We achieve this capability through judicious choice of features and feature attributes used to drive recognition, our local feature extraction techniques, and our efficient search strategy that exploits invariant feature attributes using our Vertex Space representation.

In earlier programs, sponsored by ONR and industry IR&D (Martin Marietta Labs and Schlumberger), we developed and investigated the concept of Vertex Space, an invariant representation which maps image features to a two parameter space with invariance to many viewing geometry variations. Our high level search strategy exploiting Vertex Space, was developed and its computational complexity was analyzed theoretically and with automated clutter experiments. The strategy developed for indexing and feature matching encodes model properties into a compact database by synthetically rendering views of the target, extracting features and mapping the features to Vertex Space. The Vertex Space representation for each view is stored in a hash table used for indexing by an input image analyzed in a similar way.

Indexing (rapid preliminary recognition) results were demonstrated for simple and complex real targets using target models and imagery available at that time. In addition, techniques were developed, based on implicit contours derived from orientation consistency, to extract stable features from real IR imagery for Vertex Space analysis without the need for explicit edge or region extraction [Whi95].

In our most recent ONR program, we evaluated our system performance using additional IR imagery data (from WPAFB, NVEOL and Eglin Air Force Base) and geometry models consistent with the imagery and found that sensitivity to small scale image details was limited by the nature of extracted curvature regions. Typically, Increased sensitivity to accommodate low resolution imagery results in interpreting much of the image clutter as valid features. Further, curvature regions associated with small target features often interact, resulting in regions merging into amorphous blobs. Since system performance is limited by the quality of extracted features, improved feature extraction, specifically curvature region processing, was identified as a critical need.

We developed and implemented effective curvature processing techniques, including non-linear filtering and region splitting based on curvature strength and feature orientation, which greatly improved our system's feature extraction performance. Our system is now capable of reliably resolving densely packed, interacting, small scale features. As a result, we were able to demonstrate feature matching between real IR target imagery and the corresponding model, using only target geometry information. We have not yet explicitly implemented synthetic IR signature generation, which should be considered in further investigations. Our feature matching demonstration is recent and we have not yet fully evaluated the results, but it is clear that there are some problems resulting from discrepancies between the target model and imagery, apparently due to variations of vehicles.

In the evolution of our ATR technology, the emphasis of our work has been on development of our basic ATR technology, which can serve as the basis for a complete, end-to-end ATR system. We began with the concept and development of a high-level search strategy for model to image matching. We then developed the necessary techniques for the extraction of stable and sensitive features that drive the matching process. After initial evaluation, we improved the feature extraction process and demonstrated performance consistent with a practical ATR system. The feature extraction and matching processes were developed as separate modules that can run independently or together. Taken together, they form the core of our ATR system.

Further Investigations

At the current stage of our ATR technology development, it is appropriate to re-evaluate system performance to determine the impact of recent improvements. In particular, special attention should be given to evaluating and refining the search process, since recent progress has focused more on feature extraction. We are already aware of some additional constraints that could be effectively incorporated into the indexing software module.

It is also now appropriate to develop the additional modules required to realize a complete end-to-end ATR system built upon the existing core modules, which would facilitate evaluation. The additional modules required must perform the following functions:

- determine viewing geometry from overconstrained sets of matched features
- project model features into the image using calculated geometry
- determine correspondence between projected model features and image features
- make the final recognition decision based on an appropriate set of threshold values

Once an end-to-end ATR system has been developed, it can be used to determine its capabilities by performing the necessary evaluations, with an emphasis on the high level matching and recognition processes, which, at that time, will be less developed and mature than the feature extraction processes. An initial evaluation of the complete ATR system would serve as the baseline for determination of impact resulting from subsequent improvements.

The initial evaluation would define the limits of the current treatment of IR phenomenology which does not use explicit IR modeling, but only target model geometry. As indicated by the evaluation results, current IR capabilities could be extended through a treatment of scale processing in the visible domain. Scale processing techniques may be effective in adapting visible target signatures to simulate IR signatures and may also contribute to general feature extraction stability and help to accommodate observed discrepancies between model geometry and real targets.

More fundamental means of treating IR phenomenology could also be investigated, including proprietary and commercially available synthetic IR modeling packages, such as PRISM, TTIM and IRMA, and special data sets intended for use in building model databases, such as those generated by the ARPPA UGV/RSTA project [Mun95] and NVEOL.

Synthetic imagery generation provides a theoretical framework and a high level of generality, but it may be difficult to achieve the necessary degree of signature fidelity. Real IR imagery provides realism, but is limited in generality. The current imagery was acquired using only the narrow range of conditions present at the acquisition time. It is also restricted to only a few targets and 0 degree elevation (ground level) views. A study of the different alternatives for treatment of IR phenomenology would be required to decide whether to use a synthetic imagery or real imagery approach.

Promising techniques for the generation of synthetic IR signatures, have been developed by Jonathan Mitchel who is currently at AbTech. As a graduate student at University of Virginia, Dr. Michel worked with Prof. N. Nandhakumar developing physically-based techniques for IR modeling and ATR [Mic94]. Their work advanced the state of IR modeling to a level where it could provide descriptions of IR targets sufficient for performing successful recognition, as evidenced by an Air Force program at AbTech that demonstrated a working statistical ATR concept based on training using the synthetic data provided by Dr. Mitchel's system.

An alternative to model preparation with synthetic imagery is the use of special IR image sets intended for modeling a given target. Andy Akerman and Ron Patton at Nichols Research Corporation have successfully used some of this data (generated by NVEOL and the ARPA RSTA/UGV program) to demonstrate impressive recognition results in recent ARPA Image Understanding programs [Ake96]. They use critical point features, very similar to our vertex features, for their hashing scheme and they have refined the raw data sets to optimize feature extraction for model preparation. Other potential areas for further investigation include:

- Investigate extending the ATR technology developed in the base performance period to the Synthetic Aperture Radar (SAR) domain.
- Develop techniques for treatment of articulated targets (such as tanks with moveable turrets). A likely approach would be decomposition of articulated targets into unarticulated components and consideration of constraints among the components.
- Use feature grouping based on perceptual significance to define reduced subsets of image data to drastically reduce the amount of data needed to be analyzed at one time, which would result in increased efficiency
- Implement adaptive model generation based on signature stability to improve both efficiency and stability. More signatures would be used in viewing regions where the signature changes rapidly and less where it is more stable so the modeling process adapts to the local stability of the viewing geometry.
- Extending Vertex Space to accommodate complete contours (not just contour segments in the vicinity of vertex features) may enhance both the high level matching process as well as increasing the stability of extracted features.

BIBLIOGRAPHY

[Ake96] Akerman, A., Patton, R., "Target Identification Using Geometric Hashing and FLIR/LADAR Fusion", Proceedings of the 1996 ARPA Image Understanding Workshop, Vol. 1, Palm Springs, CA.., 1996

[Mic94] Michel, J., Nandhakumar, N., "Unified 3D Models for Multisensor Image Synthesis," Proc of IEEE 2nd CAD-Based Vision Workshop, Champion, PA, February 1994.

[Mun95] Munkeby, S.H., "RSTA September 94 Data Collection", SSV Report, ARPA Order 8353, Martin Marietta Astronautics, January 31, 1995

[Whi88] Whitten, G., "Vertex Space Analysis and Its Application to model-based Object Recognition", IEEE Computer Vision and Pattern Recognition, 1988

[Whi95] Whitten, G., "Automated Missile Aim Point Selection Technology Final Report", Final Report for ONR program #N00014-92-C-0087, 1995